UNIVERSITY OF ALBERTA

Toward Human-Centered Computing: Ontological Approach to Fuzzy Intelligent Systems

by

Cuong Ly

A Thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the Degree of

MASTER OF SCIENCE

DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

Edmonton, Alberta

Fall 2007

Library and Archives Canada

Published Heritage Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et Archives Canada

Direction du Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-33300-6
Our file Notre référence
ISBN: 978-0-494-33300-6

NOTICE:
The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:
L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

Abstract

Human-Centered Computing is a methodology that focuses on human factors to make computers more human-oriented, and thus more pleasant to use. Human-Centered Computing requires computers to learn and understand human beings and work in the fuzzy way - the way human brains work.

This thesis focuses on building a fuzzy-based Human-Centered System. In this work, fuzzy concepts are used in combination with ontology in order to model information in a structured and meaningful form that is comprehensible by both human and machine. This allows human users to transfer their vague information to the system, and allows the system to "learn" more about them and implicitly derive some knowledge based on the stored information.

The last part of work presented in this thesis focuses on finding a new way of gathering user's information using a combination of natural language processing, computing with word, and ontology.

Acknowledgements

I would like to express my sincere thanks to my supervisor, Dr. Marek Reformat, for his invaluable feedback, guidance and supports, as well as for sharing with me his knowledge and experiences. He is always helpful for discussion about research and career development. I really appreciate his time and painstaking efforts for providing me detail markups for thesis outline and numerous key ideas which use throughout this research.

Special thanks to NSERC and iCORE for their financial supports, which allow me to fully concentrated in my research and acquire a number of important accomplishments.

Thanks to my wife, Trang Le, my parents, Sinh Ly and My Ca, and my sister Hoa Ly for their constant love, support. They encourage me and could be counted on to uplift my spirits and my motivation whenever I needed it.

Finally, I would like to express my appreciation to Administrative and Technical Support Staffs for their valuable supports and for providing a conducive environment in which carry out my study.

Contents

1 Introduction 1

1.1 Human-Centered Computing 1

1.2 Motivations 3

1.3 Contributions 6

1.4 Thesis Organization 7

2 Fuzzy Ontology 9

2.1 Motivation 9

2.2 Background: Semantic Web & Ontology 10

2.2.1 Overview 10

2.2.2 Ontology Languages Family 11

2.2.3 Web Ontology Language (OWL) 13

2.2.3.1 Ontology Definition 14

2.2.3.2 Ontology Instance 16

2.2.3.3 Semantic Web Rule Language (SWRL) 18

2.3 Background: Fuzzy Logic & Approximate Reasoning 21

2.3.1 Fuzzy Logic 21

2.3.1.1 Basic Definitions 23

2.3.2 Approximate Reasoning 28

2.4 Implementation Supports 28

2.4.1 Jena2 28
2.4.2 Protege & its plug-ins 29
2.4.3 OWLJessKB 29
2.4.4 Jess 30
2.4.5 FuzzyJ 30
2.5 Related Work 30
2.6 Fuzzy Ontology Framework 33
2.6.1 Overview 33
2.6.2 Fuzzy Ontology 34
2.6.3 Fuzzy Domain Specific Ontology 36
2.6.4 Application Ontology 37
2.7 Fuzzy Ontology Reasoner 39
2.7.1 Structure of the Fuzzy Reasoner 39
2.7.2 Reasoning Process 40

3 Human-Centered Service **43**

3.1 Motivation 43
3.2 Background: Semantic Web Services 45
3.2.1 Web Services 45
3.2.2 Semantic Web Service Structure: OWL-S 46
3.2.3 Implementation Supports 48
3.2.3.1 Apache Tomcat 48

3.2.3.2 Mindswap's OWL-S API 49
3.3 Related Work 49
3.4 Human Behavioral Patterns in Semantic Web Services 52
3.4.1 Human-oriented Architecture 52
3.4.2 User Acceptance Profile 55
3.4.2.1 The significant of User Acceptance Profile 55
3.4.2.2 Modeling User Acceptance with Fuzzy Ontology . . . 56
3.4.3 Human-imitating Component 61
3.4.3.1 Required Knowledge 62
3.4.3.2 Human-imitating Behavior Process 63
3.5 Example: Hotel Reservation Service 64
3.5.1 Basic Service Components 64
3.5.1.1 Hotel Information Ontology 65
3.5.2 User Information Ontology 66
3.5.2.1 User Acceptance Ontology 66
3.5.3 Rules 69
3.5.3.1 Implementation Aspects 69
3.5.4 Service Example #1 70
3.5.5 Service Example #2 72

4 Computing with Words based Systems with Fuzzy Ontology 80

4.1 Motivation 80

4.2 Background: Computational with Word by Zadeh 81
4.3 Related Work 84
4.4 Computing with Words based System 88
4.4.1 Construction of Explanatory Ontology 88
4.4.1.1 Relation Ontology 88
4.4.1.2 Constraining Ontology 90
4.4.1.3 Explanatory Ontology 91
4.4.2 Population of the Explanatory Database Ontology Instance . . 94
4.4.3 Implementation of the Computing with Words based System . 95
4.4.3.1 Architecture and Behavior 95
4.4.4 Experiments and Results 98
4.4.4.1 Explanatory Ontology 98
4.4.4.2 Known Knowledge 100
4.4.4.3 Rules 102
4.4.4.4 Part I - New Information 104
4.4.4.5 Part II - Query 104

5 Conclusions **109**

5.1 Contributions 109
5.2 Future Work 111

Bibliography **113**

Chapter 1

Introduction

1.1 Human-Centered Computing

When it first created, the computer was intended to use for only one purpose: mathematical calculations. However, its processing power and continuous development made it one of the most important invensions of the century. Nowadays, people are using computers for almost every possible activity, from writing simple text documents to controlling space shuttles. The computers have been built with a focus on technology and how humans interact with technology, rather than on how technology can be used in supporting human work. Unfortunately, researchers found that "technologies are designed around a set of assumptions concerning what work processes are required and how they will take place that are often simply wrong" [19]. Computer's speed doubles every year. Computers can now connected to each other

regardless of their geographical location. Software systems have also become bigger and are able to solve difficult problems. Yet, interacting with computers is still a problem. Since the first home computer entered the market in 1977 until now, we are still "interfacing" with computers using mice, keyboards and computer-oriented commends. Learning how to use a keyboard alone is already a challenging task. In addition to that, humans have to perform work in the ways defined by the computer that are not forgiving. If a user makes even a slightest mistake, a computer will respond with an uncomprehencable message instead of attemting to understand what he/she wants. For example, the address http://webmail.ualberta.ca will lead to an error message that is not understandable by most people. On the other hand, the correct address https://www.ualberta.ca, that has to be typed in, is not natural for humans. The error message talking about something called HTTPS that is caused by a missing letter s makes users even more frustrated and confused. It seems that it is expected that users have to have knowledge about the HTTPS protocol. Things are even worse when different software programs require different ways of interaction.

The lack of human consideration in design of both computer software and hardware tremendously increases our dissatisfaction with computers. People start questioning whether the computers serve us, or we serve them. Why do we have to make an extra effort whenever we need computers? Why do we have to interact with them the way they want? Should not it be the other way around? In the response to these prob-

lems, a new research direction called *Human-Centered Computing* has been proposed. Human-Centered Computing is not a new technology but a novel methodology that focuses on human factors to make computers more human-oriented, and thus more pleasant to use. Human-Centered Computing is closely related to Human-Computer Interaction . However, unlike Human-Computer Interaction , Human-Centered Computing recognizes that "a human is more than eye and finger movements" [19], that different persons or different groups of people have different needs; therefore, computers are required to interact with people in a natural–human–way; to understand users' needs in order to better serve users; and to use and communicate with implicit knowledge as human naturally do.

1.2 Motivations

The book *"The Unfinished Revolution: Human-Centered Computers and What They can do for us"* by Michael Dertouzos [11] introduces an ideal Human-Centered Computer which has following six main capabilities:

- Freely exchange of information.
- Get information that user want quickly and easily.
- Help people to work together.
- Talk to users using a natural language.

- Adapt to individual needs.
- Do work on behave of the user.

The first three capabilities of the Human-Centered Computer can be easily fulfilled with current technologies. For instance, Web Service technology allows computers all over the world to freely exchange information; search engines – such as Google – allow us to find information quickly; and online whiteboards help people from different geographical regions to work together. However, the last three capabilites are more difficult to achieve as they require computers to learn and understand human beings, and work in the fuzzy way – the way human brains work. The computers can not longer just work with absolute *"true"* and *"false"*, but also *"may be"* or *"partial true"*. They will no longer have all the explicit and precise data provided to them, but will also have to derive implicit knowledge from existing and sometime imprecise information. Once the computer systems understand the users, they can easily adapt to users' needs, and work on users' behalf. This motivates our vision of a Human-Centered Computing process. In this process, human user specifies his/her needs in natural language form. This information, together with stored knowledge, allows the system to learn and adapt to the user, then does work for the user. The Figure 1.1 outlines that vision.

As seen on the figure, the process contains four main components: a Speech Recognition and Natural Language Processing engine, a fuzzy knowledge base, an

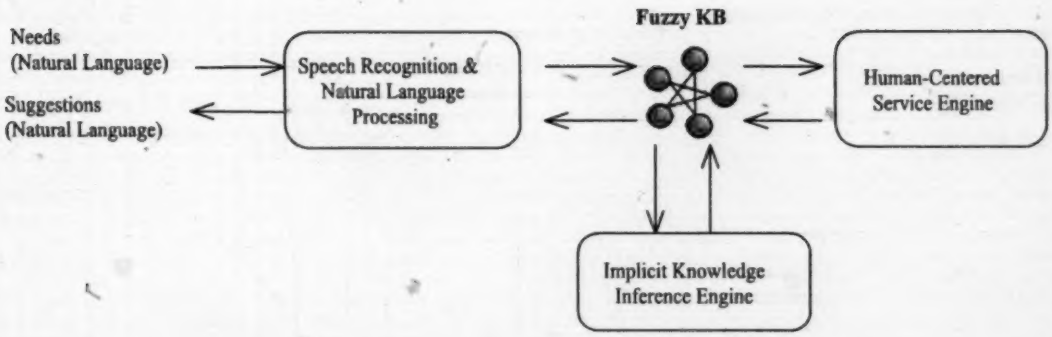


Figure 1.1: Human-Centered Computing Process

implicit information inference engine, and a Human-Centered Service engine. The speech recognition and natural language process engine converts natural language sentences into form of ontology knowledge which is structured and meaningful to the machine. The fuzzy knowledge base is the center information storage of the whole system. It contains everything the system knows, including facts and rules, and even vague or incomplete information. This knowledge base is also in form of ontology. Implicit knowledge inference engine is an inference engine that capable of putting pieces of known information together and deriving implicit information. Finally, the Human-Centered Service engine does work automatically for the user based on its understanding of the user needs.

In the work for this thesis, we focus on defining and building the last three components: the Fuzzy Ontology is the core of the fuzzy knowledge base; the computational with word based system representing the implicit knowledge inference engine; and the

Human-Centered Service.

1.3 Contributions

This thesis focuses on building a fuzzy-based Human-Centered System. In this work, fuzzy concepts are used in combination with ontology in order to model information in a structured and meaningful form that is comprehansable by both human and machine. This allows human users to transfer their vague information to the system, and allows the system to "learn" more about them and implicitly derive some knowledge based on the stored information. For example, a person who looks for a hotel room can specify that he/she would like to spend "about $150" per night. The system will understand that the user would accept a hotel room with a price of $175, but of course, not as much as one with a price of $150; and will not accept a hotel room with a price of $250. To equip the system with such capabilities a fuzzy ontology is created to represent vague and incompleted information. The system then analyzes given information and builds a complete user acceptance profile representing him/her. This process uses explicit and derived information. Based on this acceptance profile, the system find services or products that best match user's needs using Approximate Reasoning approach.

This thesis presents a fairly innovative work in the field of Human-Centered Computing as it looks for a way of representing human's behavior. The fuzzy ontology

created for this work is one of a very few practical fuzzy ontologies that exist. The concept of a user acceptance profile is a new idea that has not been seen elsewhere. The traditional way of representing user is based on a user preference profile. However, a preference profile contains only explicit information that is provided by the user. The acceptance profile includes explicit and implicit information that is used to distinguish different acceptance levels of possible choices.

The last part of work presented in this thesis focuses on defining a system capable of deriving implicit knowledge using the Computing with Word paradigm introduced by Zadeh. In this system, ontology with fuzzy concepts is used to implement the explanatory database, and fuzzy inference rules are applied for computing with word process. The system allows users to specify imprecise information using natural language terms. Based on this information, the system derives implicit knowledge to learn more about the user. For example, implicit information can be inferred when a **boy** is described as **"about sixteen year old"** is: *he is young and very likely interested in computer games.*

1.4 Thesis Organization

This thesis consists of three related but independent parts. Each part contains its own motivation, background and related work sections. The motivation sections describe why and how a particular work was performed. The background sections present the

knowledge necessary to understand and conduct the work, while the related work sections summarize the work performed by others in the field.

Chapter 2 describes the fuzzy ontology which is one of the building blocks necessary for all the other work presented in the thesis. In this chapter, the web ontology language, fuzzy concepts, and approximate reasoning are introduced in the background section. In the following sections, the fuzzy ontology framework is proposed together with the description of a fuzzy reasoner that was designed to process the fuzzy ontology.

Chapter 3 represents a Human-Centered Service Architecture. It shows how user acceptance profile is created and used in a Semantic Web Service environment. Prototype of a hotel reservation Semantic Web Service is introduced to illustrate the proposed concept.

In chapter 4, a Computing with Word based system with fuzzy ontology is presented. This chapter describes a system that is capable of gathering and representing imprecise information using natural language, then inferring implied information using the Computing with Word paradigm. An experiment showing how such system is built and how it works is included.

Chapter 2

Fuzzy Ontology

2.1 Motivation

Ontology has been widely used for information sharing and reuse, especially in the recent years with the development of the Semantic Web. The greatness of ontology lay in its capability of defining concepts and relationships between entities, and the description logic that allows software agents to process information and communicate to each other effectively. Unfortunately, the current ontology language only supports the expression of precise and crisp information while the realistic world is full of vague but useful information. Therefore, it is important that ontology is capable of expressing fuzzy information to model the real world. The Fuzzy Ontology proposed in this work allows better expression of human belief, preference, and other aspects that are important for successful human-service interaction. And more importantly,

it is designed for fuzzy reasoner to easily and efficiently process knowledge represented in the ontology.

2.2 Background: Semantic Web & Ontology

2.2.1 Overview

The concept of the Semantic Web was introduced in May 2001 in Scientific American by Tim Berners-Lee, James Hendler, and Ora Lassila [4]. Over the last few years the Semantic Web has been described in many ways: an extension of the current web in which information is given well-defined meaning, a place where machines can analyze all the data on the Web [4]. A common element of all of these definitions is a reference to a new method of representing data. The formation of the Semantic Web has been led by advances in the area of data and knowledge representation. In the nutshell, the Semantic Web can be seen as a new representation of resources on the World Wide Web. It is virtually a hub of linked information that can be accessible and operable by programs. These programs can be in a form of software agents or any other applications which are capable of handing the semantics of the information.

A new representation of resources on the web is based on usage of ontology. Ontology is a formal, explicit specification of a shared conceptualization [21]. It is a set of well-defined classes to describe data models in the specific domain. Ontology

has ability to present interrelated resources. Together with their instances, ontologies work as knowledge characters to express the individual facts [52].

2.2.2 Ontology Languages Family

In the Semantic Web environment, ontology is specified using an ontology specification language. The earliest ontology specification language is Resource Description Framework (RDF). RDF is a foundation for processing metadata [57]. RDF is a standard for describing resources and information on the web. It provides interoperability between applications that exchange machine-understandable information on the Web.

Resource Description Framework Schema (RDFS) is used as an ontology language supporting exchange of knowledge over the web. RDF and RDFS serve as the basic methodology of expressing web resources in the form of triples: resource, subject, and property.

A later ontology specification language is a combination of DARPA Agent Markup Language (DAML) and Ontology Inference Layer (OIL). Its name is DAML+OIL. It enables the creation of ontologies for any domain and the instantiation of these ontologies in the description of specific web sites. DAML+OIL enhances and extends RDFS with richer modeling primitives [3] to represent the semantics of resources and information.

The latest web resource ontology language is Web Ontology Language (OWL), which is the recommendation by World Wide Web Consortium (W3C). OWL is not only for representing information on the web, but it also improves the capability to process the information and increases the interoperability among software agents [38].

A structure representing hierarchy of markup languages used for representing resources on the web is shown in Figure 2.1 [39]. Section 2.2.3 will give more details of OWL as it is mainly used in our work. A detail and formal specification of RDF and DAML+OIL can be found at http://www.w3.org/RDF/ and http://www.daml.org.

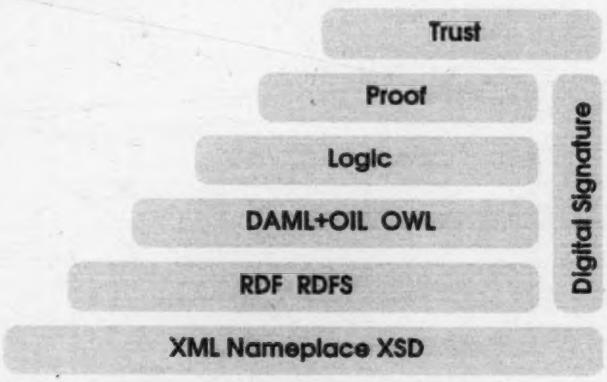


Figure 2.1: The architecture of the Semantic Web

2.2.3 Web Ontology Language (OWL)

As mentioned in Section 2.2.2, OWL is the latest and official ontology language recommended by the W3C due to its capability of facilitating "greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDFS) by providing additional vocabulary along with a formal semantics" [66]. OWL has three sub-languages including OWL Lite, OWL DL and OWL Full with the expressiveness increases from OWL Lite to OWL Full. The complexity, however, increase as the expressiveness increase. So it is depending on the application that an OWL language should be used. OWL Lite is suitable for applications that need a classification hierarchy and simple constraints while OWL DL and OWL Full should be used in ones that require maximum expressiveness. OWL DL is often more preferable than OWL Full as it retains computational completeness (all conclusions are guaranteed to be computable) and decidability (all computations will finish in finite time). The only advantage of OWL Full over OWL DL is that it allows syntactic freedom of RDF (for example, an identifier can be used as both class name and instance name).

As space is limited, we will only introduce the basic features of OWL language that we use in this work. For a complete specification of OWL, please visit http://www.w3.org/TR/owl-ref/.

2.2.3.1 Ontology Definition

The ontology definition layer represents a framework used for establishing a structure of ontology and for defining concepts existing in a given domain. A structure of ontology is built based on a relation *isa* between concepts. This relation represents a *subClassOf* connection between a superclass concept and a subclass concept. In such a way, a hierarchy of concepts is built. This hierarchy is a partially ordered set of concepts, and the resulted ontology is a directed acyclic graph.

Additionally, the ontology definition contains detailed descriptions of all concepts of ontology. Concepts are defined using two types of the properties: datatype properties, and object properties. Both of them provide a way for an accurate and complete description of a concept. The details of both types of properties are presented below:

- datatype property - this type of property focuses on describing features of a concept, datatype properties are used to represent attributes of the concept that can be express as values of such data types as boolean, float, integer, string, and many more (for example, byte, date, decimal, time);
- object property - this property defines more sophisticated, comparing to *isa*, relationships among concepts (nodes), these relationships follow the notion of Resource Description Framework (RDF) [25] that is based on a triple *subject-predicate-object*, where: *subject* identifies what object the triple is describing; *predicate* (property) defines the piece of data in the object a value is given to;

and *object* is the actual value of the property; for example, the triple "John lives in Edmonton" has "John" as subject, "lives in" as predicate and "Edmonton" as object.

Both types of properties are very important for defining an ontology. The possibility of defining concept attributes and any relations between concepts creates a very versatile framework capable of expressing complex situations with sophisticated concepts and multiple relationships of different kinds.

An example of ontology definition is shown in Fig. 2.2. A simple concept *ED: Person*[1] illustrates all aspects of ontology definition. The hierarchy is represented by a relationship *isa*: the concept *ED:Student* is a subclass of the concept *ED:Person*, and at the same time it is a superclass for the concepts *ED: ElementarySchoolStudent*, *ED:JuniorHighSchoolStudent*, *ED:HighSchoolStudent*, and *ED:UnversityStudent*. Each of the concepts is described by a number of properties. The datatype properties of the concept *ED:Person* are *ED:lastName* of type string, and *ED:age* of type float. As we can see, most of the properties that defined the concept *ED:Person* are object properties[2]. These properties define relationships that exist between the concept *ED:Person* and other concepts defined in the ontology. The object properties

[1]The prefix *ED* has been assigned to this ontology at the time it was imported to Protege - a tool used for ontology construction. Other prefixes that can be seen in the paper are: *f:*, *f_constraint:*, *namebook:*, *product:*, *school*, *location*.

[2]The object property has a keyword "Instance" in the description of a node, Fig. 2.2.

ED:hasChild and *ED:hasParent* indicate that an instance of *ED:Person* can have a relationship *ED:hasChild* with another instance of the *ED:Person*, and a relationship *ED:hasParent* with yet another instance of the *ED:Person*. Another object property *ED:liveIn* links an instance of *ED:Person* with an instance of the concept *ED:Location* that could be *Edmonton* or *London*, or any other city that is an instance of the concept *ED:Location*. Other properties include:

- *ED:likes* that allows for representing what a person who is "defined" by the concept *ED:Person* likes - in this case, the links will lead to instances of the concept *product:Product*;
- *ED:gender* provides an information about a gender of an instance, the ontology *namebook* contains the concept *namebook:Gender*, and an instance of it (two instances are *male* and *female*) can be associated with the instance of *ED:Person*;
- *ED:firstName* links *ED:Person* with one of instances of the concept *namebook:Names* - this allows for indicating a first name of the *ED:Person*.

2.2.3.2 Ontology Instance

Once the ontology definition is constructed, its instances can be built. It means that the properties of the concepts are filled out with real data - values are assigned to datatype properties, and links to instances of other concepts are assigned to object properties. An example of ontology instance is presented in Fig. 2.3. The center of the

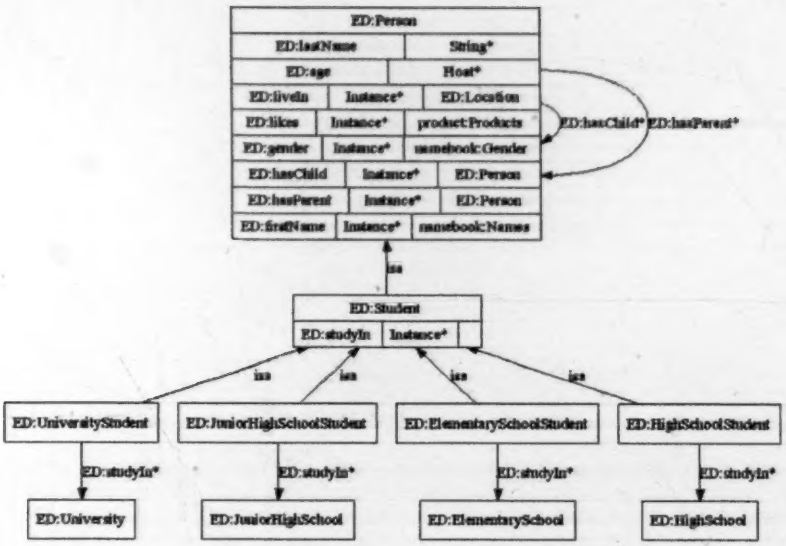


Figure 2.2: Example of Ontology Definition

graph is occupied by the entry *John* that is the instance of *ED:HighSchoolStudent*[3]. *John* has a number of relationships with other instances: the relationship *ED:studyIn* with the instance *QueenElizabeth*, and the relationship *ED:firstName* with the instance *namebook:John*. The *namebook:John* is the instance of the concept *namebook:Names*, and by itself it has a relationship *namebook:forGender* with the instance *namebook:Male*. The instance *John* also has a datatype property *ED:lastName* equals to *Smith*, *ED:age* equals to *16*.

[3]An instance of a given class is identified by a connection *io* between an instance and its definition.

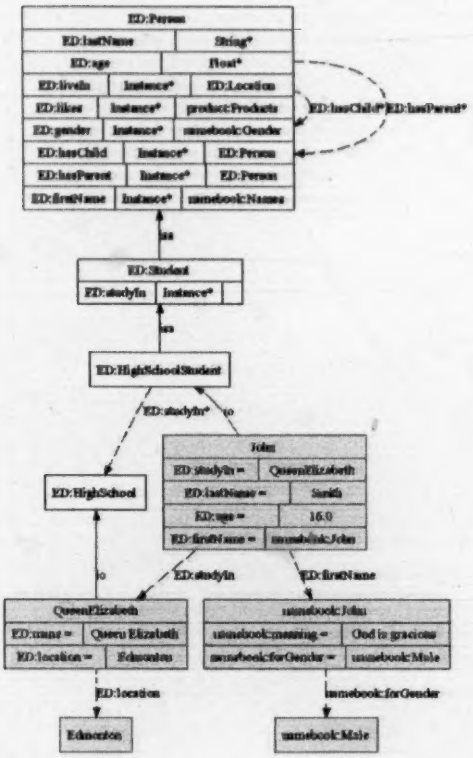


Figure 2.3: Example of Ontology Instance (white boxes represent definitions of concepts, gray boxes represent instances)

2.2.3.3 Semantic Web Rule Language (SWRL)

It has been identified [28] that the OWL has limitations in the case of representing relations between complex properties. This has been overcome by putting together OWL and a rule language. As the result of that, the Semantic Web Rule Language (SWRL) has been introduced [28] [46] as a combination of OWL with RuleML (the sub-language of Rule Markup Language).

In SWRL, a rule axiom consists of an antecedent (body) and a consequent (head). The basic element of both antecedent and consequent is an atom. SWRL defines five basic atoms that can be used to build a rule:

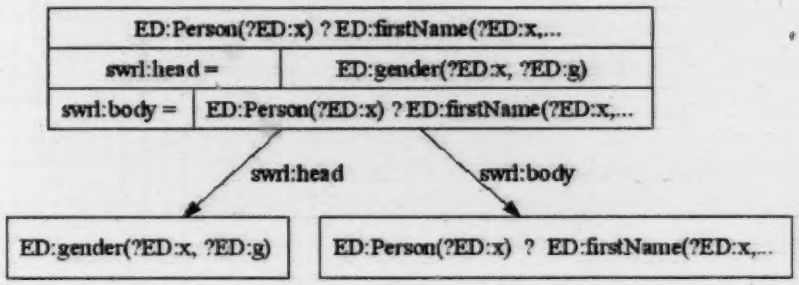
- *C(x)* it is the simplest atom, it is used to check if a given instance x is the instance of concept C, for example, $Person(John)$ represents an atom that checks if $John$ is the instance of the concept $Person$;
- *P(x,y)* it is the atom that allows for checking if two instances x and y are related to each other via a property P, for example, $liveIn(John, Edmonton)$ is "looking" at the existence of the property $liveIn$ between the instances $John$ and $Edmonton$;
- *Q(x,z)* it is the atom that verifies if a data property Q of instance x has a value z, for example, $lastName(John, Smith)$;
- *sameAs(x,y)* holds if instances (individuals) x and y are the same;
- *differentFrom(x,y)* holds if instances (individuals) x and y are different;

All atoms presented above can be used with variables instead of instances (x, y) and values (z). In this case, the atom $P(x, y)$ can be used in the following way - $liveIn(?a, Edmonton)$, and it would represent a question: who lives in Edmonton?

Using the SWRL together with ontology, it is possible to build rules based on object properties of the concepts defined in this ontology. An example of such a rule is presented in Fig. 2.4.

```
ED:Person(?ED:x) &
ED:firstName(?ED:x, ?namebook:n) &
namebook:forGender(?namebook:n, ?namebook:g)
-> ED:gender(?ED:x, ?namebook:g)
```

(a)



(b)

Figure 2.4: Ontology-based Rule: (a) its informal "human readable" form, (b) and RDF graph form

The first atom of the rule is *ED:Person(?ED:x)* of the type *C(x)*, and it leads to selection of all instances of the concept *ED:Person*. The following atom is *ED:firstName(?ED:x,*

?namebook:n) that is of the type *P(x,y)*. An object property *ED:firstName* of the concept *ED:Person* (Fig. 2.3) is used here, and it defines the triple < *ED:Person, ED:firstName, namebook:Names* >. Therefore, this atom returns as the result a set of tuples $< x, y >$ where x represents a person from the set of persons identified by the first atom, and y is an element of the set of names. The third atom *namebook:forGender(?namebook:n, ?namebook:g)* (type *P(x,y)* is the triple < *namebook:Names, namebook:forGender, namebook:Gender* >. It finds out how many of the instances of *namebook:Names* are in the relation with the instances of the concept *namebook:Gender* (there are only two instances here: *namebook:male* and *namebook:female*). In other words the antecedent finds, based on a person's name, if the person is female or male. The consequent of the rule is the atom *ED:gender(?ED:x, ?namebook:g)* that "links" instances of the concept *ED:Person* to the instances of the concept *namebook:Gender*.

2.3 Background: Fuzzy Logic & Approximate Reasoning

2.3.1 Fuzzy Logic

Real situations are very often not crisp and deterministic and they cannot be described precisely. They are very often uncertain or vague in a number of ways. One of uncertainty is related to lack of information about the future state of the system.

This type of uncertainty is handled appropriately by probability theory and statistics. It is assumed that the events are well defined. This is in contrast to the vagueness concerning the description of the semantic meaning of the event, phenomena or statements, which is called fuzziness [76] [88] [47].

Fuzziness can be found in many areas of daily life. It is particularly frequent, however, in all areas in which human judgment, evaluation, and decisions are important. One of the most important reasons for that is that human daily communication uses natural languages and a good part of human thinking is done with it. In these natural languages the meaning of the words is very often vague. The meaning of a word might even be well defined, but when using the word as a label for a set, the boundaries within which objects belong to the set or not, become fuzzy or vague. Examples are words such as "birds" (how about penguins, bats, etc.?), "red flowers", but also terms such as "tall men", "creditworthy customer". In this context, two kinds of fuzziness with respect to their origins can be distinguished: intrinsic fuzziness and informational fuzziness. The former is illustrated by "tall men". This term is fuzzy because the meaning of tall is fuzzy and dependent on the context. An example of the latter is the term "creditworthy customer": a creditworthy customer can possibly be described completely and crisply if a large number of descriptors is used. However, this is more than a human being could handle simultaneously. Therefore the term, which in psychology is called a "subjective category" becomes fuzzy.

The idea of fuzzy theory was first introduced by Lotfi Zadeh of the University of California at Berkeley in the 1960s [75]. Zadeh was working on the problem of computer understanding of natural language. Natural language is not easily translated into the absolute terms of "true" and "false". Fuzzy logic includes "true" and "false" as extreme cases of truth about phenomena or statement. Fuzzy logic also includes the various states of truth in between. For example, the result of a comparison between two things could be not "tall" or "short" but "0.38 of tallness."

2.3.1.1 Basic Definitions

A classical (crisp) set is normally defined as a collection of elements of objects $x \in X$ which can be finite, or countable. Each single element can either belong to or not belong to a set A, $A \subseteq X$. In the former case, the statement "x belongs to A" is true, whereas in the latter case this statement is false. Another way of describing it can be done by defining the member elements by using the characteristic function, in which "1" indicates membership and "0" non-membership. For a fuzzy set, the characteristic function allows various degrees of membership for the elements to a given set.

Definition 1: If X is a collection of objects denoted generically by x then a fuzzy

set A in X is a set of ordered pairs:

$$A = (x, u_A(x)) \parallel x \in X) \tag{2.1}$$

u_A is called the membership function or grade of membership of x in A which maps X to the membership space M. The range of the membership function is a subset of the nonnegative real numbers whose supremum is finite. Elements with a zero degree of membership are normally not listed.

Example 1: A person wants to classify her willingness to book a hotel room. One indicator of it is the acceptance level of a hotel room price. Let X = {\$50, \$100, \$150, \$200, \$250} be the set of possible room prices described by x = price. Then the fuzzy set "acceptance of a hotel room based on its price" may be described in the following way

$$A = \{(\$50, \mathbf{1}), (\$100, \mathbf{0.8}), (\$150, \mathbf{0.6}), (\$200, \mathbf{0.3}), (\$250, \mathbf{0.1})\}$$

As it can be seen, the level of acceptance is expressed as a number in the range [0.1, 1]. The highest acceptance level is "1" for a room price of \$50, and the lowest acceptance level is equal to "0.1" for a price of \$250.

Within the last 30 years numerous special notations of fuzzy sets have been suggested. One of them – linguistic variables, is of a special interest.

Definition 2: A linguistic variable is characterized by a quintuple

$$(x, T(x), U, G, M) \tag{2.2}$$

in which x is the name of the variable; $T(x)$ is the term set of x, that is, the set of names of linguistic values of x, with each value being a fuzzy variable defined on U; G is a syntactic rule for generating the names of values of x; and M is a semantic rule for associating with each value its meaning. A particular x, that is a name generated by G, is called term.

Example 2: If *room price* is interpreted as a linguistic variable, then its term set $T(room\ price)$ could be

$$T = \{very\ cheap,\ cheap,\ moderate,\ expensive\}$$

where each term in $T(room\ price)$ is characterized by a fuzzy set in a universe of room price [\$0, \$250]. *Very cheap* can be interpreted as a room price below \$50.00, a room price about \$100 as *cheap*, a room price about \$160 as *moderate*, and a term *expensive* will be identifying a price of above \$250. These terms can be characterized as fuzzy sets whose membership functions are shown in Figure 2.5.

For example, the fuzzy truth that *room price* takes on the value *cheap*, also denoted by (x *is cheap*), is designated by a fuzzy value in square brackets. If the price of a room is \$120, then the fuzzy truth is 0.45. This can be written:

(x *is cheap*)[0.45]

The room price (of \$120) may have positive fuzzy truths of membership in multiple fuzzy sets. It may be that

$$(x \textit{ is cheap})[0.45] \; AND \; (x \textit{ is moderate})[0.15]$$

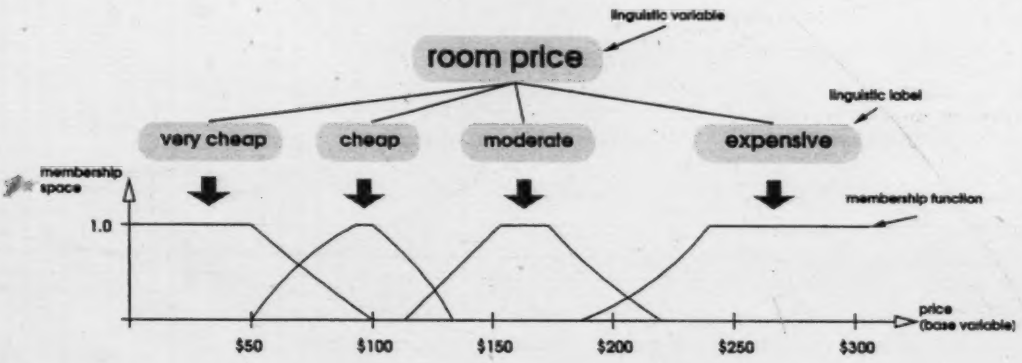


Figure 2.5: The example of linguistic variable

Obviously the crucial element of a fuzzy set is the membership function. Methods have been designed to determine membership functions empirically. Often they are computed on the basis of formal assumptions. It is important to realize which type of membership function is available. In some cases, it is sufficient to know the grades of membership of the element of discrete fuzzy sets, which cannot be interpolated. In other cases it is absolutely necessary to know the membership function analytically.

Definition 3: A fuzzy preposition is a condition or statement (in English or symbols) that takes a fuzzy truth value. A new proposition can be formed by the ANDing, ORing, or NOTing of other propositions. An atomic fuzzy proposition is one that

can not be decomposed into two or more simpler fuzzy propositions (it has a form such as (x *is cheap*)).

Definition 4: A fuzzy rule is a proposition that contains a main implication that divides it into a left hand side and a right hand side, where the left hand side contains one or more conjuncted fuzzy propositions and the right hand side contains an atomic fuzzy proposition.

Example 4: Let us define three linguistic variables: *room price*, *hotel location*, and *acceptance*. The terms of these variables are:

$$T(\textit{room price}) = \{\textit{very cheap},\ \textit{cheap},\ \textit{moderate},\ \textit{expensive}\}$$

$$T(\textit{hotel location}) = \{\textit{close},\ \textit{far}\}$$

$$T(\textit{acceptance}) = \{\textit{low},\ \textit{moderate},\ \textit{high}\}$$

Based on that, the following proposition can be composed:

$$(\textit{room price is cheap})\ AND\ (\textit{location is far})$$

and the following fuzzy rule:

$$((\textit{room price is cheap})\ AND\ (\textit{location is far})) \Rightarrow (\textit{acceptance is moderate})$$

As it can be seen, usage of fuzzy sets, linguistic variables and fuzzy rules bring an enormous capability to express opinions and statements in more natural, human way with semantic vagueness.

2.3.2 Approximate Reasoning

Additionally, approximate reasoning, whose basic principles have been formulated by Zadeh [77] [78], is reasoning in the presence of incomplete or inaccurate information[4]. An inference engine built based on this principle is able to process meanings rather then symbols. Such scenario can be applied to modeling of human reasoning. In such case, the response to a given request can be not only "true" or "false" but also an approximation. An important advantage is that approximate answers can often be more meaningful than fully correct answers [64].

2.4 Implementation Supports

2.4.1 Jena2

Jena2 is a Java framework for developing Semantic Web applications. it provides an API for creating ontology in either RDF or OWL, and supports SPARQL, the query language for RDF. Jena2 is developed by the HP Labs and is free of charge. More information on Jena2 is found at http://jena.sourceforge.net/.

[4]Approximate reasoning is based on fuzzy Modus Ponens. A detailed description of approximate reasoning based on fuzzy logic can be found in [12] [13].

2.4.2 Protege & its plug-ins

Protege is a graphical Ontology editor for modeling and building ontology. It provides a plug and play environment that allows different tools to work together for rapid prototyping. Some of the useful plug-ins for Protege includes:

- **Protege-OWL**, the core of the ontology editor. It is built-in Protege since version 2.0. Protege-OWL is built on top of Jena and provides an API that can be used as an alternative to Jena2 API.
- **OntoViz**, a visualization tool for visualizing ontologies. It uses a drawing software called Graphviz from AT&T. So we will need Graphviz install if we want to use the OntoViz plug-in.
- **SWRLTab**, allows user to build ontology with SWRL rules and even execute the rules by integrating a Jess engine). SWRLTab also provides a SWRL-JessBridge API that allows SWRL rules together with OWL knowledge to be translated into Jess knowledge.

2.4.3 OWLJessKB

OWLJessKB is a bridge that translate an OWL ontology into Jess format. DAML-JessKB (the previous version of OWLJessKB) is one of the first tool that allows ontology knowledge to be used with an inference engine. However, its limitation is the translated knowledge is not very easy to used; so when SWRLTab came out

with a SWRLJessBridge API, it became a more preferable choice of translating OWL ontology into Jess knowledge.

2.4.4 Jess

Jess is a very efficient Java inference engine for the rule-based system inside which Rete algorithm is used [16]. Jess API allows user to directly manipulate and reason about Java objects. Jess is created and supported by Sandia Labs. Academic version of Jess is available free of charge.

2.4.5 FuzzyJ

FuzzyJ is a collection of Java classes for handling fuzzy concepts and reasoning in a Java environment. FuzzyJ has been developed by National Research Council of Canada's Institute for Information Technology [40]. FuzzyJ can be used either by itself or by integrating with other rule engines such as CLIPS or Jess. In this work, the combination of FuzzyJ and Jess has been used to create a powerful and efficient fuzzy reasoner.

2.5 Related Work

When the project is started in May 2003, there was very few work done on fuzzy ontology. Today, there are countless number of publications on this research field which proves that there is a high demand for fuzzy ontology in the Semantic Web

world. Currently, the research on fuzzy ontology divides into two main directions: Automatic Generation of Fuzzy Ontology and Fuzzy Ontology Definition.

The formal direction focuses on finding fuzzy relations between ontology concepts from text based documents using data mining techniques such as clustering or classification. Some of the popular work on this direction are the fuzzy ontology for news summarization by Lee, Jian and Huang[36], and the fuzzy ontology generation framework by Tho, Q.T. et. al [61]. In the work done by Lee, Jian and Huang, meaningful terms from Chinese news in some websites are collected then classified according to events of the news. Based on the classification of terms, a fuzzy membership is assigned to each event to represent how it related to a specific concept. A fuzzy ontology was created to capture all the generated fuzzy relations. The fuzzy ontology is then used by an agent to summarize the news. Similar to the work done by Lee, Jian and Huang, Tho, Q.T. et. al propose a framework called FOGA (Fuzzy Ontology Generation frAmework) which is used to automatically generate fuzzy concept hierarchy for certain domains such as citation database. In this framework, uncertain information is clustered into conceptual clusters using fuzzy conceptual clustering technique. An fuzzy hierarchy of concepts is then generated automatically using formal definition of the concepts.

Different with the first direction of Semantic Web fuzzy ontology research, Fuzzy

Ontology Definition direction focuses on defining a standard way of representing fuzzy information to be used in any application domain. The fuzzy ontology in this direction is usually created manually to describe the core fuzzy logic definitions and fuzzy relations. Using this fuzzy ontology, vague information can be expressed through fuzzy logic terms which can be directly processed and manipulated by any fuzzy reasoning engine. The work done by Straccia [59], Gao and Liu [18], and Stoilos et. al. [58] are some of the popular examples. Straccia and Stoilos et. al. work are quite similar in the sense that they both propose a fuzzy version of $\mathcal{SHOIN}(\mathcal{D})$, the formal definition of Description Logic in OWL DL, call f-$\mathcal{SHOIN}(\mathcal{D})$. In their work, they show how $\mathcal{SHOIN}(\mathcal{D})$ concepts can be represented using fuzzy logic definition such as t-norm, t-conorm, negation and implication. Stoilos et. al. go a step further by propose a reasoning procedures for f-$\mathcal{SHOIN}(\mathcal{D})$ and how f-$\mathcal{SHOIN}(\mathcal{D})$ can be mapped into a fuzzy version of OWL, called f-OWL. The works of Straccia and Stoilos et. al, however, are mainly the mathematically proofs of concepts. No specific fuzzy OWL feature is syntactically defined. Not going deep into description logic level of OWL ontology, Gao and Liu take a simple approach by showing how fuzzy membership values can be represented by using the formal OWL language.

Our work follows the later direction in which we manually create fuzzy ontology to represent vague information. Similar to work done by Gao and Liu, we use formal OWL language to describe fuzzy logic concepts. However, unlike their work, our fuzzy

ontology is capable of not only representing fuzzy membership value, but also express fuzzy terms using fuzzy membership functions. More importantly, our fuzzy ontology is designed in a specific way to best suite our reasoning engine, which depends on FuzzyJ library to reasoning about vague information.

2.6 Fuzzy Ontology Framework

2.6.1 Overview

As mentioned before, our Fuzzy Ontology is built to allow vague information to be expressed using core fuzzy logic concepts. Therefore, it is important to understand that the proposed Fuzzy Ontology is not much useful when used by itself as when it is used with other ontology to describe fuzzy concepts in that ontology. The Figure 2.6 shows framework, in which the Fuzzy Ontology should be used. As seen in the Figures, the core of the Fuzzy Ontology framework is the Fuzzy Ontology which is capable of describing any fuzzy linguistic in term of fuzzy membership function. Built on top of the Fuzzy Ontology is Fuzzy Domain Specific Ontology which is an ontology that describes specific fuzzy concepts by using the fuzzy ontology. The top layer is the application ontology which can be any ontology that use the fuzzy concept in the Fuzzy Domain Specific Ontology.

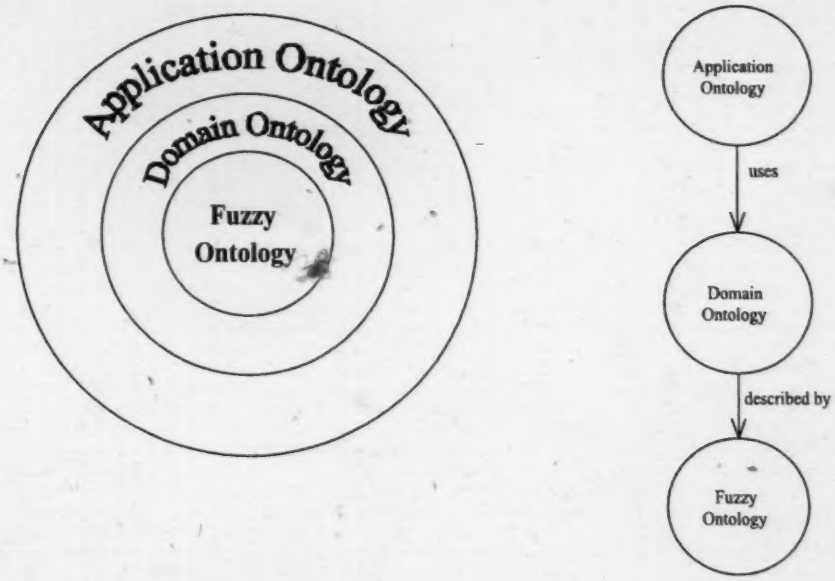


Figure 2.6: Fuzzy Ontology Framework

2.6.2 Fuzzy Ontology

As illustrated in Figure 2.7, core fuzzy logic concepts that are related to fuzzy information expression, such as fuzzy variable, fuzzy terms, membership functions, as well as their relationships, are captured in the Fuzzy Ontology in order to provide a uniform and structured means of fuzzy knowledge representation. In this ontology, fuzzy membership functions type S, Π and Z are defined using fuzzy pairs. Specifically, membership function type S and Z are described by two fuzzy pairs: start point and end point. Start point defines the value where the curve start and the membership value at that point. Membership function Π is different in which it has only one

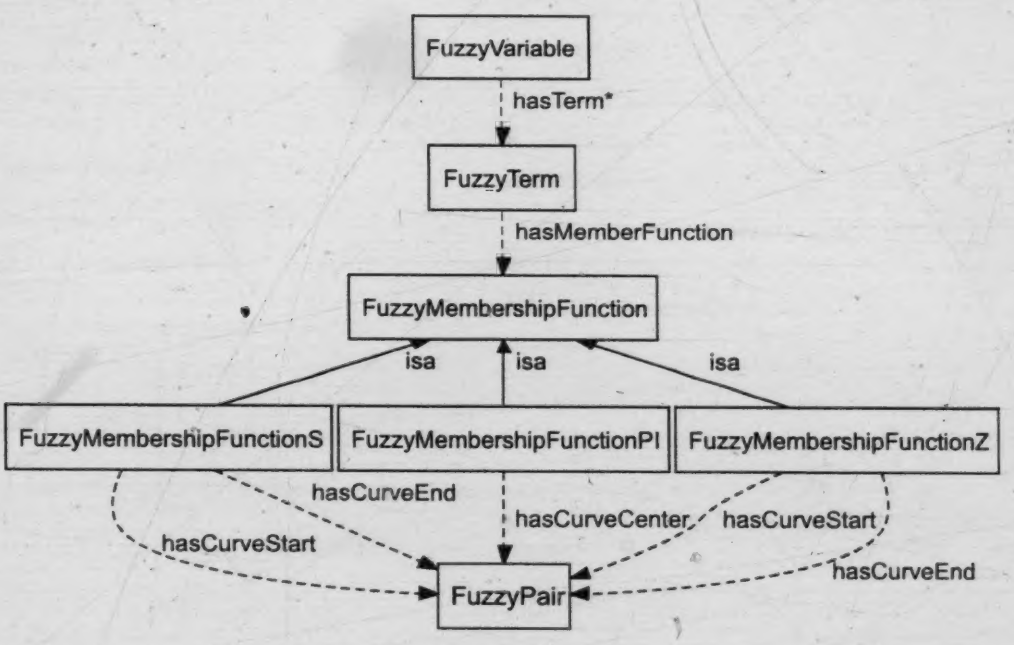


Figure 2.7: The core of the Fuzzy Ontology

fuzzy pair to describe the middle point and a double value that define the width of the curve. Figures 2.8, 2.10 and 2.9 visualize how these functions are described.

The Fuzzy Ontology is in compliance with OWL DL, that means it supports computational completeness and decidability of reasoning systems. It guarantees that reasoning systems would be able to compute all conclusions in finite time. It is also worth noting that, since the ontology is ultimately consumed by the fuzzy reasoner, its concepts and properties are designed in the way which closely resembles the representation of fuzzy concepts in FuzzyJ, one of the main component of the fuzzy reasoner.

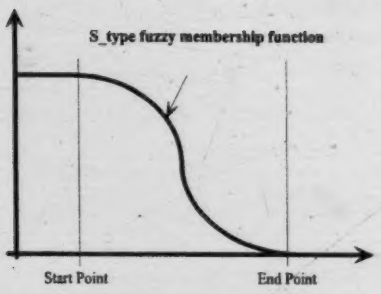


Figure 2.8: Fuzzy membership function Z

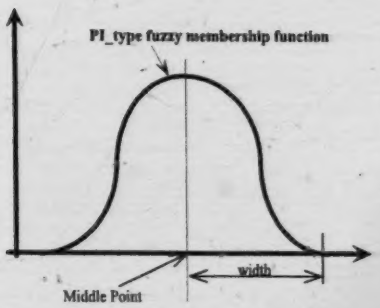


Figure 2.9: Fuzzy membership function Π

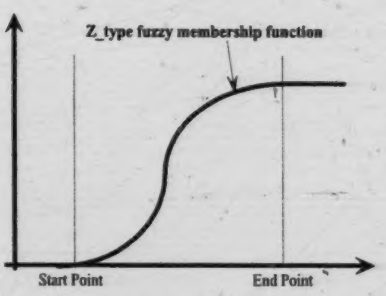


Figure 2.10: Fuzzy membership function S

2.6.3 Fuzzy Domain Specific Ontology

Fuzzy Domain Specific Ontology is an ontology that describes domain specific concepts using fuzzy linguistic. An example of such ontology is Price Ontology (Ontology that describes prices) shown in Figure 2.11. In the Price Ontology, prices can be described in fuzzy linguistic terms as "very cheap", "cheap", "moderate", "expensive" or "very expensive". In order for machines to understand the meanings of these terms, the Price Ontology has to describe the terms using fuzzy functions that are defined in

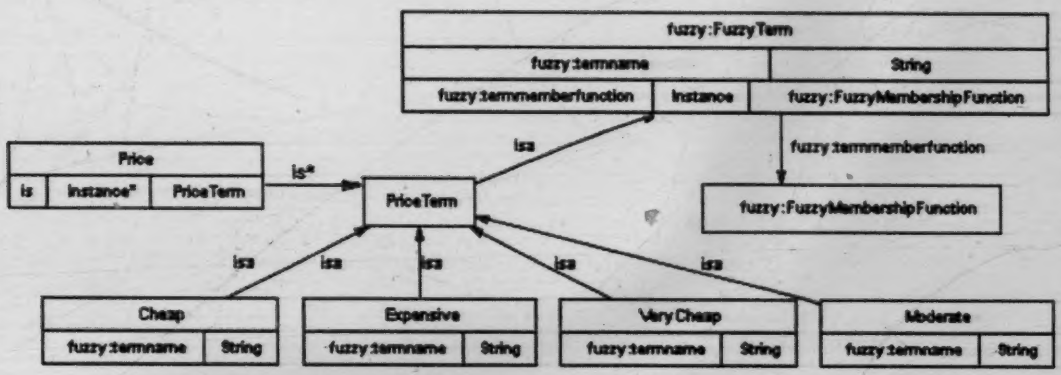


Figure 2.11: Price Ontology

the Fuzzy Ontology. Once it is defined, we can use the Price Ontology to describe a specific type of price; for instance, the hotel room price in example 2.3.1.1. The RDF graph and the OWL code showing how Cheap Hotel Price is expressed are presented in Figure 2.12 and Figure 2.13.

2.6.4 Application Ontology

Application Ontology is the top layer ontology that is directly used by an application, for instance, an ontology that describes user preferences about a certain product that can be used by an application to collect user's feedback about the product, by a search engine to find more accurate the information or by a human-oriented service such as the one proposed in chapter 3 to better serve its users. One simple example of preference ontology is ontology describes hotel price reference where a user specifies what he/she means by cheap, moderate or expensive hotel room price using the Price

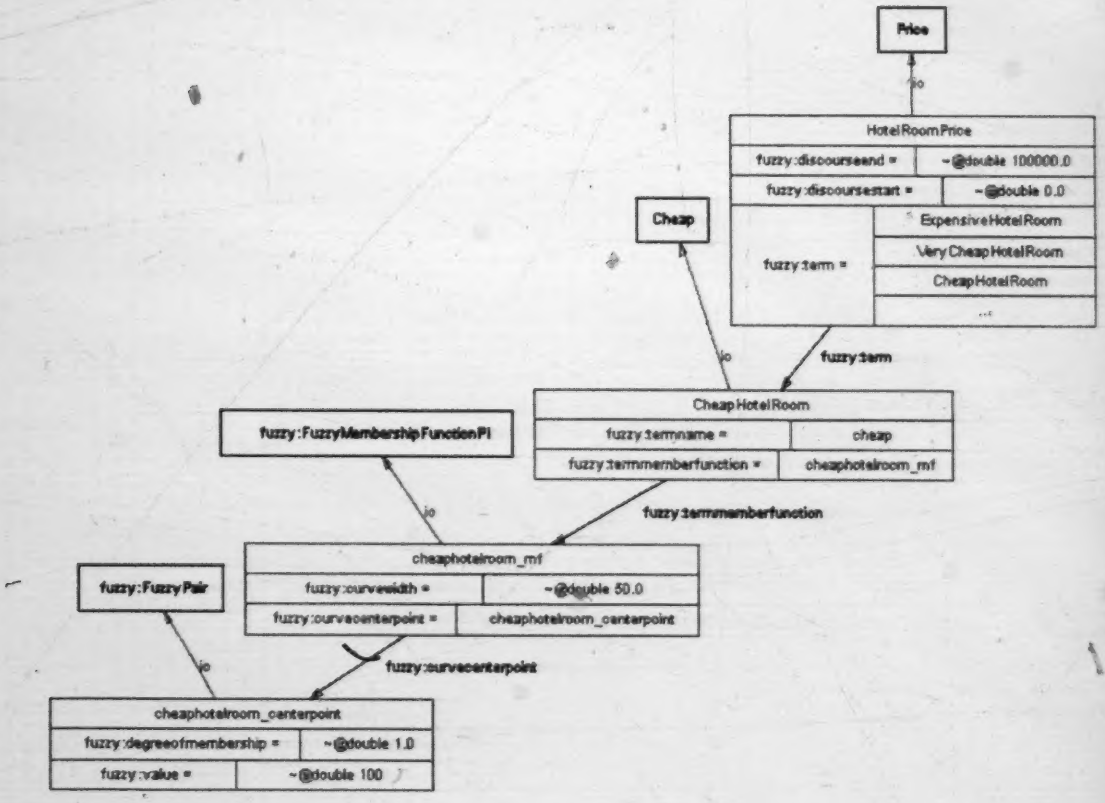


Figure 2.12: RDF graph expression of Cheap Hotel Room Price

Ontology above.

```
<Price rdf:ID="HotelRoomPrice">
 <fuzzy:term rdf:resource="#ExpensiveHotelRoom"/>
 <fuzzy:term rdf:resource="#VeryCheapHotelRoom"/>
 <fuzzy:term rdf:resource="#CheapHotelRoom"/>
 <fuzzy:discoursestart rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
 >0.0</fuzzy:discoursestart>
 <fuzzy:term rdf:resource="#ModerateHotelRoom"/>
 <fuzzy:discourseend rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
 >100000.0</fuzzy:discourseend>
</Price>

<Cheap rdf:ID="CheapHotelRoom">
  <fuzzy:termmemberfunction>
    <fuzzy:FuzzyMembershipFunctionPI rdf:ID="cheaphotelroom_mf">
      <fuzzy:curvewidth rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
      >50.0</fuzzy:curvewidth>
      <fuzzy:curvecenterpoint>
        <fuzzy:FuzzyPair rdf:ID="cheaphotelroom_centerpoint">
          <fuzzy:degreeofmembership rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
          >1.0</fuzzy:degreeofmembership>
          <fuzzy:value rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
          >100</fuzzy:value>
        </fuzzy:FuzzyPair>
      </fuzzy:curvecenterpoint>
    </fuzzy:FuzzyMembershipFunctionPI>
  </fuzzy:termmemberfunction>
</Cheap>
```

Figure 2.13: OWL expression of Cheap Hotel Room Price

2.7 Fuzzy Ontology Reasoner

2.7.1 Structure of the Fuzzy Reasoner

As shown in Figure 2.14, OWLJessKB is on the top layer for the reasoner acting as a translator to convert OWL into Jess knowledge. The core of the reasoner is the combination of Jess and FuzzyJ with three set of rules: OWL specific rules, fuzzy information builder rules and application rules. OWL specific set of rules comes with the OWLJessKB. They are used to translate implied information from OWL specific constructs such as inheritances, transitive property, functional property, hasValue restriction and so on. Fuzzy information builder rules, on other hand, are specific for Fuzzy Ontology. They are used to convert fuzzy information described by the

Fuzzy Ontology into FuzzyJ representation. Lastly, the application rules are application dependent that define how the reasoner should process the available information.

Note that the term "fuzzy inference engine" is sometimes used as an interchangeable term for the fuzzy reasoner.

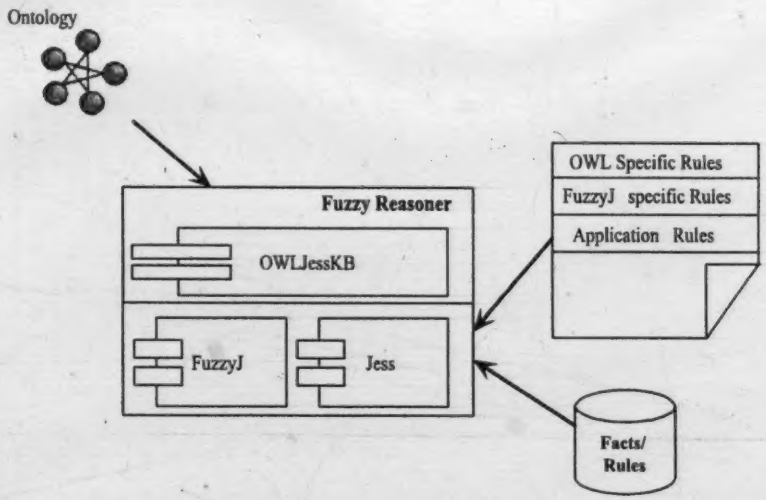


Figure 2.14: Structure of Fuzzy Reasoner

2.7.2 Reasoning Process

The reasoning process is started with the translation of OWL ontology and instances into Jess facts using the OWLJessKB. The reasoner then invokes the fuzzy information builder rules to gather and build the FuzzyJ representation of fuzzy information described by the FuzzyOntology. Specifically, all instances of fuzzy variables and

fuzzy terms and instances of their subclasses in the Fuzzy Ontology are represented as FuzzyJ's fuzzy variables and terms objects. All the instances of fuzzy membership functions classes in the Fuzzy Ontology are represented as FuzzyJ's fuzzy membership functions objects. Once all the knowledge preparation step is completed, the reasoner reads the application specific rules and performs an approximate reasoning process accordingly to generate one or more useful, approximate answer(s). A graphical representation of the reasoning process is shown in Figure 2.15.

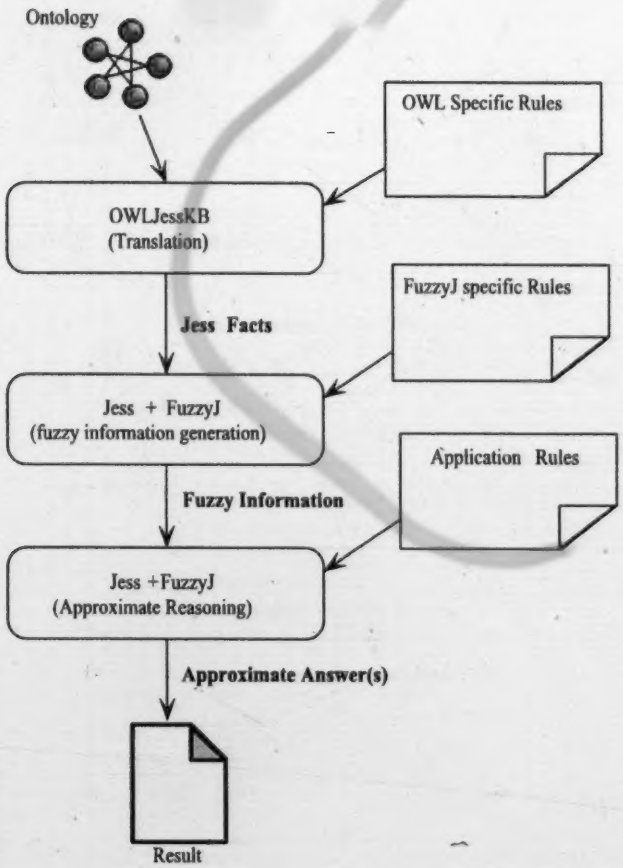


Figure 2.15: Reasoning Process

Chapter 3

Human-Centered Service

3.1 Motivation

Increased autonomy of agents means increased involvement of agents in processes of evaluating responses coming from different service providers, and making decisions regarding selection of a single best service. In order to keep the Semantic Web promise, these actions should be performed in the same way as a user would do it. Each agent representing the user in the environment of Semantic Web Services should be able to imitate user's behavioral patterns. Ability of an agent to do so will have an enormous impact on the successful outcome of the service request.

It is essential for an agent to "know" its users. Any information which characterizes the user is valuable. It can be a set of opinions, statements, patterns of behavior,

preferences, things she likes, things she tries to avoid. More information of such nature means better chances that the agent can perform its tasks in a way similar to a human being. At that point, it can be said that a model of human behavior has to be built. The model can be developed based on statements and opinions describing the user. These statements can be obtained directly from the user or extracted from information representing user's activities [32] .

However, information the user provides to her agent can be inherently imprecise. The human user would be more accurate when she could express herself in a natural way – using natural language. It would be much easier for the user to use English terms like *small*, *large*, *close*, *far*, *very much*, or *probably* to express her statements and opinions. A solution to that problem is application of fuzziness.

A need to create a "softer", more human friendly way of exchange information between a user and an agent, and dealing with this information in a more human-like way have become the driving force behind introduction of fuzziness to agents being a part of the Semantic Web environment. The application of fuzzy concept provides the capabilities for:

- better capturing of human specific semantic imprecision of opinions and statements, ensuring direct interface to agent and a chance for the agent to "see" things in a human-like way;

- better imitating human way of thinking and decision making via application of fuzzy-based reasoning.

All this is possible because fuzzy set theory has the potential for natural language discourse from the user to the computer. The theory can be viewed as a bridge between the precise milieu which the computer requires and the ambiguous world in which most problems exist. Gupta states [23]: "Fuzzy set theory is an attempt to remove "linguistic" barriers between humans, who think in fuzzy terms and machines that accept only precise instructions".

3.2 Background: Semantic Web Services

3.2.1 Web Services

Generally speaking, services could be in any type of response or action according to request. Figure 3.1 gives the scenario of a web service. Services usually involve two components: service requester and service provider. In the case of web, services can be provided in text, multimedia, and/or raw data format. The traditional service providers assume that they are dealing with human. There is a gap between request-response talk, which is handled mainly by the user. XML Web Services communicate using Simple Object Access Protocol (SOAP) and use Web Services Description Language (WSDL) to describe their capabilities in the form of XML Schema (XSD). The services providers must publish their services, and the user has to explore and find

such services manually. In this case, the user has more initiative than the services provider.

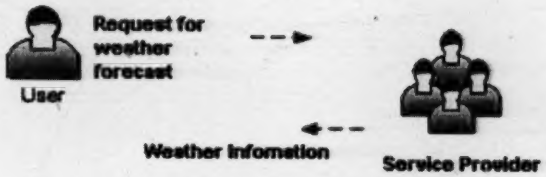


Figure 3.1: Weather Web Service

To improve the interoperability, many business to business (B2B) and business to customer (B2C) applications are used as the routing bridge between users and providers. Those applications are called web crawler because they know the structure of the certain webpages. The crawlers understand what the user wants and what the website can provide. They work perfectly in the static annotation webpage inside which contents are mostly represented in the fixed style. The drawback of such solution is that any change of service format could result in unsuccessful service.

3.2.2 Semantic Web Service Structure: OWL-S

Web resources can be presented more precisely and intensively. The contents on the website are not only accessible and understandable by both machine and human, but also by web services. In order to augment the interoperability, Semantic Web Services

can be used in such circumstances. The semantic encoding of resources, properties, objects and interfaces makes them understandable by machines. The introduction of XML Web Services has greatly enhanced an interaction between distributed applications. However, human still needs to discover the related Web Services and knows their profile before using them. As shown in Figure 3.2, the Semantic Web Services is addressing description - ServiceProfile, process - ServiceModel, composition, and grounding of Web Services what makes the service available for software agent exploitation. This is a combination of web services with knowledge representation, utilizing the ontology concept. OWL Service (OWL-S) can describe the Semantic Web Services by using OWL.

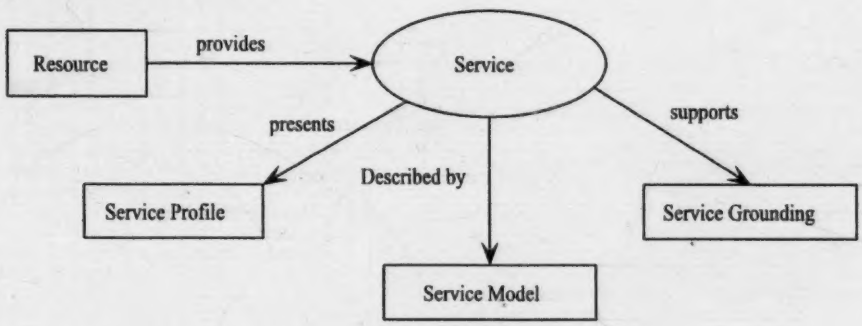


Figure 3.2: Upper Ontology of Semantic Web Services

Figure 3.3 shows the interaction between Semantic Web Services. A single Semantic Web Service can locate others services based-on their ServiceProfiles in the registry. After that, all services are able to interact with each other through the

ServiceModel and ServiceGrounding.

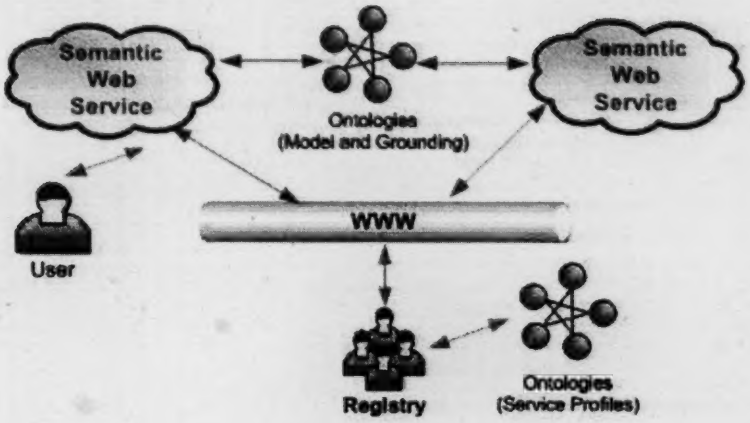


Figure 3.3: Upper Ontology of Semantic Web Services

3.2.3 Implementation Supports

3.2.3.1 Apache Tomcat

Tomcat is an open source middle-ware framework that helps developers to create web applications quickly and easily. Tomcat implements Sun Microsystems's Java servlet and the Java Server Pages (JSP) specifications which provide an environment for Java applications to run in cooperation with a web server. In this work, we use Tomcat as the web container for our Hotel Reservation system. More information about Apache Tomcat is found at http://tomcat.apache.org/

3.2.3.2 Mindswap's OWL-S API

Mindswap is one of the largest research group of people working with Semantic Web technology at the University of Maryland Institute for Advanced Computer Studies. OWL-S API is one of their active project that provides a Java API for read, write and execute OWL-S service descriptions. OWL-S API is used in this work for experimenting how Semantic Web Service works including how service profile, service process model and grounding are created from an XML web service, and how it is used by a Semantic Web Service client. More information about this OWL-S API can be found at http://www.mindswap.org/2004/owl-s/api/

3.3 Related Work

As web applications move closer to users to provide users the ease of use and the ease of finding information, getting and taking advantage of human behavior and preference become one of the major concerns. Many successful commercial websites including Amazon.com and chapters.indigo.ca learn about their user preferences through the item that the users are looking at, then using this preference to recommend other items that like-minded people are also interested in.

Applications concerning user preferences often faces the challenge of choosing most effective algorithm to achieve the best result. Attempting to improve the quality of

item recommendation, Reategui et. al [50] propose a modified version content-based method, which uses record-like structures, called *item descriptor*, to model the relationships between user features and items. The strength of a relationship is calculated using the conditional property $P(d|e)$, which represent the probability that an user having characteristic e would prefer an item having description d. For example, the probability that a business man would prefer a business book is higher than the probability that he would prefer an art book. Once the conditional probability of the relationships are established, the system calculates score for each item as an aggregation of the conditional property and ranks them before recommend to the user. This method gives out better quality recommendation if more information about user is known.

Taking advantage of user preference, Glover et. all [20] introduce a new search strategy that uses user preference to improve the precision of their meta-search engine. In this strategy, user specifies a keyword query and an information-need category. The meta-search engine will then send the keyword query to each individual search engine including Google, Yahoo, HotBot and Alta Vista. The resulted documents come from all search engines will then be ranked based on an utility function that takes the provided category in account.

The use of user preference is also applied in new research areas such as compo-

sition of web service. To automate the selection of web service composition process, Agarwal and Lamparter [35] uses fuzzy IF-THEN rules to model user preference in order to get good approximations of desired information. The user preference is then matched with aggregation information about web services using approximation approach to find the best way to combine the web services.

Traditionally web applications model user preferences in the way that is most efficient to process. For example, Reategui et. al [50] use database record to model user preference while Agarwal and Lamparter [35] use fuzzy IF-THEN rules. However, the user information stored in these structures are not easy to understand, shared and extended; for instance, the database built for the recommend system in Reategui et. al [50] would not be easily shared with or extended by other systems that do not concern about the conditional probability of the relationships between user features and item descriptions. To overcome these limitations, Razmerita et. all [49] proposed an user modeling architecture based on the promising semantic web technology. In this architecture, user profile is stored in form of ontology and its instants. The advantages of this approach are: (1) user's information is well-structured and has well-defined meaning that allows web applications to integrate with this architecture easily; (2) user preference can be easily shared and event extended by different applications.

In our work, we also represent user's profile in a similar way to Razmerita et.

al; however, the ontology we use is capable of representing fuzzy information, which is closer to human real-life preferences. The user preference is then transformed into a human component which is modeled by fuzzy functions and fuzzy IF-THEN rules to support approximate reasoning process for dealing with lack of information, uncertainty and ambiguity.

3.4 Human Behavioral Patterns in Semantic Web Services

3.4.1 Human-oriented Architecture

Combining the concept of Semantic Web Services with aspects of fuzziness provides capabilities for developing agents equipped with abilities for human-like decision making. Services can be pushed to a higher level when these agents become an integral part of an agent system implementing services. These agents are capable of making sophisticated decisions when they obtain responses from all service providers involved in a given service request. The agents have to decide which responses would be selected by the user and which of them are closest to the user's preferences. In order to make such decisions, the agents have to know what the user would do in the exact or similar circumstances. This implies a need for knowledge about the user. The knowledge about the user possessed by the agent should include:

- preferences – information what the user wants, what service aspects are impor-

tant for her, what particular things she is looking for;

- acceptance levels – representation of user's willingness to scarify her preferences, i.e. how comfortable she is when preferences are partially satisfied or not satisfied at all;
- decision rules – statements regarding user's approaches to making decisions in a number of different circumstances.

Natural language is the best form of providing and representing this information. Application of fuzzy approach allows for usage of such terms like *small*, *large*, *low*, *high*, and statements like *if price is high then acceptance of this hotel is low*. These terms and statements express user's requirements and her way of dealing with different responses to a service request. Additionally, usage of approximate reasoning ensures that this information is used for human-like reasoning.

In order to perform decision making processes with information described above a special human-oriented architecture of the Semantic Web Services is proposed. A diagram of the architecture is presented in Figure 3.4. The architecture contains the following components directly related to realization of the Semantic Web Services: a OWL-S Port and Service [43]. The OWL-S Port consists of three modules. The *OWL Parser* is used to load OWL-S specification of the web service. It transforms OWL files into list of predicates to be processed by the *OWL-S Virtual Machine*. The OWL-S Virtual Machine defines a knowledge base that implements the OWL-S

Service Model semantics. The *Web Service Invocation* module transforms information to be sent to other Web services into concrete messages. When the OWL-S Port is responsible for the interaction with other Web services, the Service part controls what the Web service does. In such case, this part should have an Agent Control Component representing Web service actions. At the same time, there is a need for Human-imitating Component. This component represents human as a user of given service. Its main role is to mimic human behavior. Its detailed description is presented in Section 3.4.3.

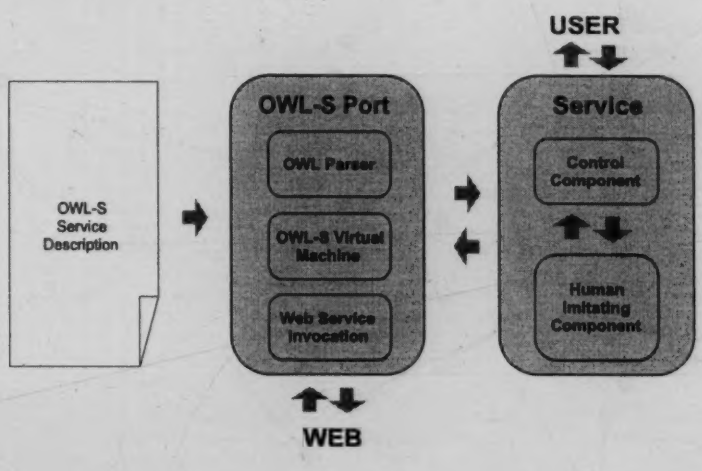


Figure 3.4: The human-oriented architecture of a user agent

3.4.2 User Acceptance Profile

3.4.2.1 The significant of User Acceptance Profile

It has been realized that user preference obtained by simple queries poorly represents human behavior which is often vague and complicate. For instance, an user specifies that she would like to find a hotel room that cost from $100 USD to $200 USD per night. A human immediately understand that a hotel room with price of $95.99 USD is as preferable as a room with price of $110 USD, which is more preferable than on with price of $180 USD. However, a computer software would return all the hotel rooms that have price range from $100 USD to $200 USD with equally favorableness and ignores one with price of $95.99 USD. To overcome this limitation, a User Acceptance Profile is proposed. The User Acceptance Profile is built from a careful analysis of user preferences. It applies fuzzy concepts to represent user preferences based on acceptance levels. On the User Acceptance Profile, the hotel room with price of $100 USD would have acceptance level of 1.0; the one with prices of $95.90 USD and $110 USD would have equal acceptance level of 0.95, while the one with price of $180 USD would have acceptance levels of 0.7. This means that the hotel room with price of $100 USD is most preferable, then ones with prices of $95.99 USD and $120 USD. The room with price of $180 USD is the least preferable.

3.4.2.2 Modeling User Acceptance with Fuzzy Ontology

User Acceptance is modeled using the Fuzzy Ontology introduced in the chapter 2. Three linguistic terms used in every acceptance factor are *low*, *moderate* and *high*. Depending on the factor it describes, each of the term is dynamically expressed using a fuzzy membership function with associated fuzzy pair(s). The user acceptance ontology has to be specific for each type of service. In the case of our Hotel Reservation Service example, the Acceptance Ontology including user's acceptance regarding hotel rooms, services, and facilities is represented in Figure 3.5.

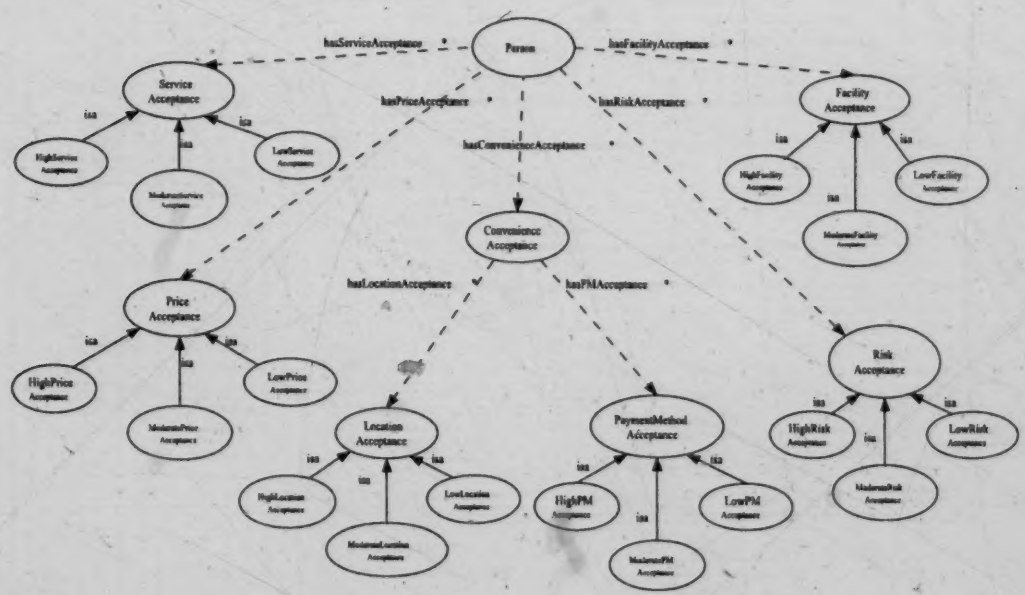


Figure 3.5: The User Acceptance Ontology

An important step of an ontology construction process is determination of parameters defining membership functions associated with linguistic terms for each fuzzy variable. Two different approaches can be applied:

- the first approach is based on direct involvement of a user; he/she is asked about specific information that is directly applied to set up parameters of membership functions;
- the second approach focuses on defining membership functions indirectly – the parameters of the functions are derived from the data provided by a user.

Examples of both cases are given below in the case of the Hotel Reservation Service. The first approach can be applied for such aspects as *price acceptance* and *localization acceptance*. A user provides values representing limits of the domain and "boarders" between linguistic values *high acceptance* and *moderate acceptance*, as well as between *moderate acceptance* and *low acceptance*. Based on simple information provided by a user – he/she answers a couple of simple questions – parameters of membership functions are derived.

In the second approach, the user is involved in the process of creation of membership functions indirectly. This can be illustrated with an example of finding membership functions representing *service acceptance* and *facility acceptance*. A user has to make choices regarding his/her needs for service and facility items. The user can identify that he/she *must have* a specific item, or that it is *nice to have* it,

or that he/she *doesn't care* about it. Each choice is associated with a weight: it is $weight_{must}$ for *must have*, $weight_{nice}$ when the user indicates that it is *nice to have* an item, and $weight_{dont}$ when the user *doesn't care* about an item. The choices made by a user together with weights are used to define membership functions. The template for these functions is shown in Figure 3.6. The values of three important points are calculated in the following way. The value of Min is assumed zero. The formula for Mid is $Mid = \sum_i^N weigth_{must} * must_choice_i$ where N is a number of all items, and $must_choice_i$ is equal to *one* if a user has selected item i as *must have*, or *zero* otherwise. The value of Max is calculated using the formula $Max = \sum_i^N weigth_{nice} * nice_choice_i + Mid$. Similar as before $nice_choice_i$ is equal to *one* if a user has selected this item as *nice to have*, or *zero* otherwise. It is possible that none of the available items has been identified as *must have*. The equations for Max and Mid are different: $Max = \sum_i^N weigth_{nice} * nice_choice_i$ and $Mid = \frac{Max}{2}$. Once the values of these three points are known, the parameters of three membership functions can be derived assuming that the crossing points, Figure 3.6, are $\frac{Min+Mid}{2}$ and $\frac{Mid+Max}{2}$.

Of course, if all items have *don't care* choice selected then a given acceptance is not considered.

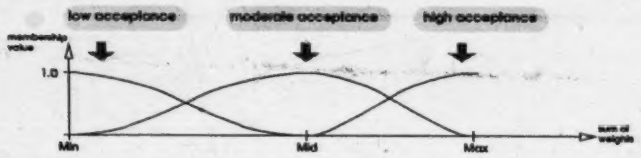


Figure 3.6: Universe of discourse and a membership functions template used for facility and service preferences

Example 5: In order to illustrate how acceptance is modeled using the Fuzzy Ontology, recall the example of Hotel Accommodation Service. One of the information which user has to provide is degree of acceptance on an available resource according to a specific preference. Human acceptance is far from being a crisp statement. For acceptance on the location of a hotel, is 1.5km from hotel to the city centre close or far? The answer depends on individual user. If the user likes walking along the street, she thinks that it is convenient concerning the location. On the other hand if the user does not have time for walking she may think it is too far. Even for the same person, the meaning of the term *close* is quite vague. If 0.5 km is close what about 0.6 km?

In this case, in order to express her acceptance level, the user would use terms *high*, *moderate* or *low* with fuzzy borders between each term. Figure 3.7 shows a portion of ontology representing the membership functions for three levels of hotel location acceptance: HighAcceptance, ModerateAcceptance and LowAcceptance. The OWL

code representing user's High Location Acceptance is shown below.

```
<HighAcceptance rdf:ID="High_LocationAcceptance">
    <fuzzy:termmemberfunction>
      <fuzzy:FuzzyMembershipFunctionZ rdf:ID="High_LocationAcceptance_mf">
        <fuzzy:curveendpoint>
          <fuzzy:FuzzyPair rdf:ID="High_LocatioinAcceptance_endpoint">
            <fuzzy:degreeofmembership rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
            >0.0</fuzzy:degreeofmembership>
            <fuzzy:value rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
            >2.0</fuzzy:value>
          </fuzzy:FuzzyPair>
        </fuzzy:curveendpoint>
        <fuzzy:curvestartpoint>
          <fuzzy:FuzzyPair rdf:ID="High_LocationAcceptance_startpoint">
            <fuzzy:value rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
            >1.0</fuzzy:value>
            <fuzzy:degreeofmembership rdf:datatype="http://www.w3.org/2001/XMLSchema#double"
            >1.0</fuzzy:degreeofmembership>
          </fuzzy:FuzzyPair>
        </fuzzy:curvestartpoint>
      </fuzzy:FuzzyMembershipFunctionZ>
    </fuzzy:termmemberfunction>
  </HighAcceptance>
```

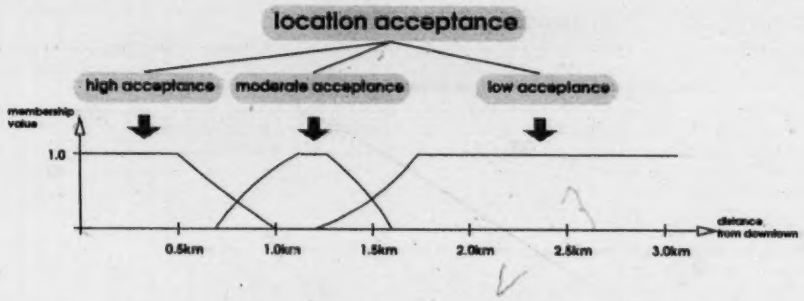


Figure 3.7: Hotel location acceptance

3.4.3 Human-imitating Component

The essential part of the proposed human-oriented architecture is Human-imitating Component. It embraces necessary elements for representing a user during her interaction with the Web service. The user component performs its tasks combining responses from different service providers and information from user-related ontologies using approximate reasoning. Each of the responses is evaluated and its acceptance level is reasoned about. A comparison of these acceptances leads to identification of the best response.

As any software component, the Human-Imitating Component has two essential parts: data and process. The data uses in the component is not simple text but ontologies and rules; therefore, it is referred as knowledge. The process is also significant as it intimates human behavior. A detailed structure of the component is presented in Figure 3.8.

3.4.3.1 Required Knowledge

The knowledge used by the component is divided into four categories: User Preference, User Acceptance Profile, Acceptance Rules and Responses from service providers:

- *User Preference* is the information provide directly by the user regarding her personal information and her preference about the service or product she wants to get. The user preference is stored in an user information ontology.

- *User Acceptance Profile* models the user's favorableness levels of each preferred domain. It is inferred from user preference and captured in the user acceptance ontology introduced in Section 3.4.2.2.

- *Acceptance Rules:* can be either obtained directly from the user or inferred from user preference to present user decisions in the case of different results obtained from service providers. The first case allows user to specify exactly what she wants. For instance, the user would specify that if her price acceptance level for a response is low, then her over all acceptance for that service is low, no other factors need to be considered. In the second case, the user would just specify which factor is important (or not important) to her, the system will then derive a set of rules to reflex her over all acceptance accordingly.

- *Responses from Service Providers:* are the responses returned regarding of the user request. These responses are matched against the user acceptance profile in order to her acceptance level for each of the factor.

3.4.3.2 Human-imitating Behavior Process

The process of imitating human behavior regarding decision making is preformed by the fuzzy reasoner introduced in Section 2.7. A number of preparation steps are preformed before the reasoning process starts. These steps include analysis of preferences and acceptance. Results of these operations together with details regarding a hotel obtained from a service provider are used as inputs to the reasoner.

Preference analysis uses parts of ontology which are related to needs and requirements of the user. Its main task is to use user's information and create the user acceptance profile. The purpose of this process is to prepare a suitable environment for comparison of service responses with user preferences. A number of different acceptance domains can be created for a given kind of service. For example, for Hotel Accommodation Service these could be: price, location, facilities, provided services.

The main role of **acceptance analysis** is to conclude if the user is willing to accept given response from a service provider. The match between user's needs and a response can result in *high*, *moderate* or *low* acceptance. This process is preformed for each acceptance domain separately. The results obtained for each domain are used to reason about the overall acceptance level of a given service. The acceptance level obtained for each domain is an input to the fuzzy inference engine. As the result, a single value identifying a degree of matching user needs to a given service provider.

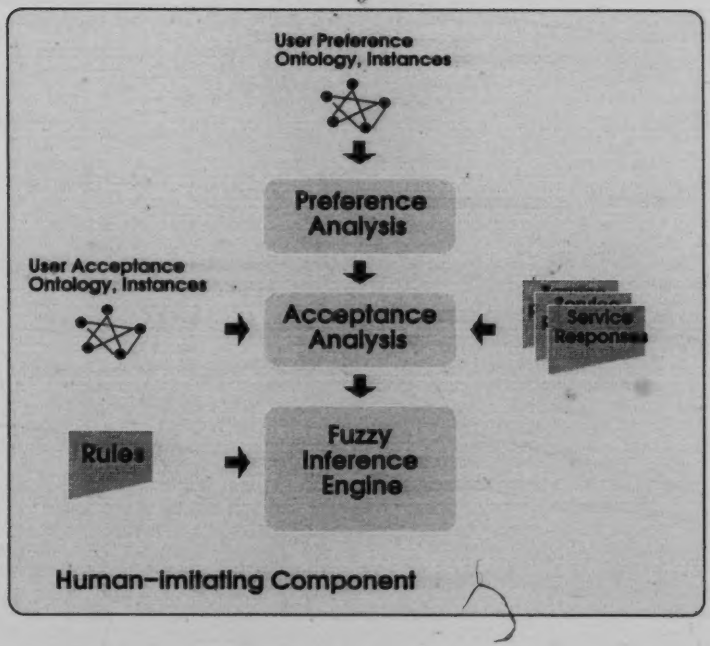


Figure 3.8: The architecture of Human-imitating Component

3.5 Example: Hotel Reservation Service

3.5.1 Basic Service Components

The capability of imitating human behavioral patterns using fuzziness is presented here for the case of Hotel Reservation Service. An agent representing a user has been implemented using described human-like architecture. In order to make this implementation possible, three ontologies have been constructed:

- Hotel Information Ontology (HIO) is a partially ordered set of all terms and concepts describing a hotel, Figure 3.9. Its instances contain every piece of information that is needed to reason about goodness of a given hotel: its local-

ization, its services and facilities, and prices for different rooms.

- User Information Ontology (UIO) that defines terms and concepts regarding a user and her preferences, Figure 3.10. This ontology is used to express information about user's requirements regarding localization of a hotel, its services, rooms and facilities.

- User Acceptance Ontology (UAO) contains specifications of terms needed to perform approximate reasoning about compliance of responses of service providers with users needs. Terms defining fuzzy linguistic labels and parameters of their membership functions representing user's opinions are part of this ontology presented in Figure 3.5. Its instance contains information what a user "thinks" about discrepancy between her requirements and responses from service providers and how she "treats" these differences.

3.5.1.1 Hotel Information Ontology

A process of making decision which hotel is the best match to the user's needs has to be done in the presents of information about the hotels. For that purpose the Hotel Information Ontology has been created. This ontology is presented in Figure 3.9. The examples #1 and #2 presented in the Section 8, use two instances of this ontology. These two instance, i.e. the values of the most relevant variables, are shown in Table 3.1.

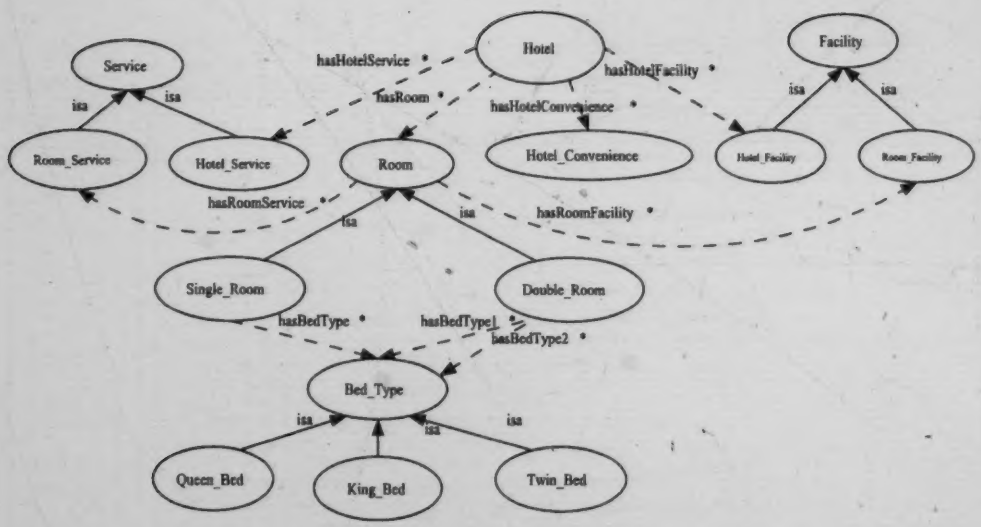


Figure 3.9: The Hotel Information Ontology

3.5.2 User Information Ontology

All information related to a user, it means personal data as well as his/her preference regarding a specific service are represented using a special User Information Ontology, Figure 3.10.

3.5.2.1 User Acceptance Ontology

The reasoning about the most fitted response among all responses obtained from service providers is performed based on four inputs: a room price, hotel localization (a distance from a hotel to the user's point of interest), services items provided by a hotel, and facilities available at a hotel. In order to make the reasoning, four linguistic (fuzzy) variables expressing user's "reaction" to different values of

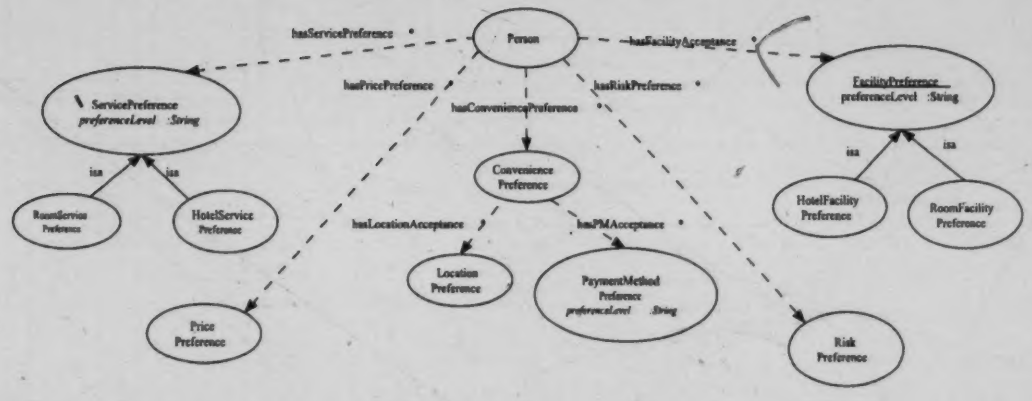


Figure 3.10: The User Information Ontology

the inputs are defined: *price acceptance*, *location acceptance*, *service acceptance* and *facility acceptance*. Each of these variables contains three linguistic terms: *high acceptance*, *moderate acceptance*, and *low acceptance*. These acceptances reflect user's willingness to agree to the response from a specific service provider. Comparison of inputs representing data about a given hotel with defined linguistic variables leads to identification of fuzzy truths used for reasoning purposes. The parameters of membership functions associated with each linguistic variable are determined using the procedure described in Section 3.4.2.2. In the example, the values of the weights are: $weight_{must} = 6$, $weight_{nice} = 2$, and $weight_{dont} = 0$.

Example 6: Let's take a look at the procedure of constructing the membership functions based on information provided by a user regarding needed facilities. A user identifies the following choices about facility items, Table 3.2. Using the equations

from Section 3.4.2.2 and the values of the weights provided above, it can be calculated that the *Max* is equal to 21, the *Mid* point is equal to 12. Such selection of items results in the membership functions presented in Figure 3.11.

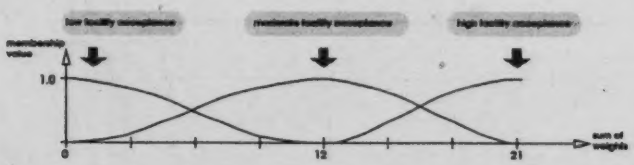


Figure 3.11: Example of automatically created membership functions for facility acceptance

Example 7: A set of choices preformed by the user regarding different service items is presented in Table 3.3. Such selection results in the following scenario: the *Max* is set to the value of 12, and *Mid* point is set to 6. The membership functions of the service acceptance are shown in Figure 3.12.

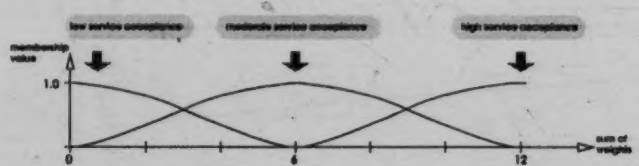


Figure 3.12: Example of automatically created membership functions for service acceptance

3.5.3 Rules

The reasoning process depends on rules provided by a user. If only two linguistic variables are considered, for example *price acceptance* and *localization acceptance* (such a case in illustrated in the Example #1 – see below) a relatively simple set of rules is needed. A table with a set of rules, called also a rule matrix, is shown in Table 3.4. For the case of four input system, the Example #2, the rules are more complex. Examples of such rules are: if **location acceptance** is *moderate* and **price acceptance** is *high* and **facility acceptance** is *low* and **service acceptance** is *high* then **service acceptance** is *moderate*, or another rule: if **location acceptance** is *low* and **price acceptance** is *high* and **facility acceptance** is *high* and **price acceptance** is *high* then **service acceptance** is *high*.

3.5.3.1 Implementation Aspects

The Human-imitating Component has been built according to the described architecture. The essential element of the component that deals with imprecise information is the fuzzy reasoner introduced in Section 2.7.

The following scenario has been adopted. All service providers available on the Web are registered in the registry. The agent queries the registry and receives a list of service providers capable of performing a hotel reservation task. It is assumed that a number of service providers have been already performed the hotel reservation tasks.

In such case, the user agent deals with a number of service responses. The agent compares these responses against preferences identified by the user. This comparison means loading ontologies and instances related to the user and hotels into an approximate reasoner. Additionally, a set of rules describing user's acceptance of services based on different degrees of price, localization, facility and service acceptances is loaded to the reasoner. The output of the reasoner represents ratings of hotel reservation responses. The result is routed to the user.

Using this implementation, a set of experiments has been conducted to gain confidence of usability of the service. Two of such experiments are presented below.

3.5.4 Service Example #1

The first example illustrates benefits brought by application of fuzziness. For this particular example only location and price acceptances are considered. A user provides the data which defines the acceptance levels. The parameters defining membership functions in the domains of price preference and location preference are presented in Figure 3.13.

The detailed information about two hotels used here is shown in Table 3.1. The comparison of data related to localization of hotels, and room prices with acceptance levels identified by the user lead to a very interesting conclusions. The graphical

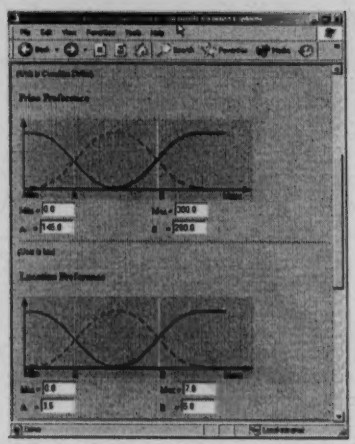


Figure 3.13: Dialog box for User's Acceptance Levels: for price (a), for location (b) (solid thin line – high acceptance, dashed line – moderate acceptance, solid thick line – low acceptance)

representation of this comparison is shown in Figure 3.14.

It can be observed that a room price of the hotel #1 is lower than a room price of the hotel #2, Table 3.1. Moreover, the price of room from hotel #2 is larger than "the boarder price" identified by the user. However, the scenario is quite different for the localization of the hotels. In this case hotel #2 is located closer to the city's centre than hotel #1. Looking at the values of room prices and localization of hotels, and values provided by the user regarding splitting of price and distance from city's centre between "high" and "moderate" it can be said that if the crisp intervals were used to define acceptance levels the hotel #2 would not be considered for booking – the

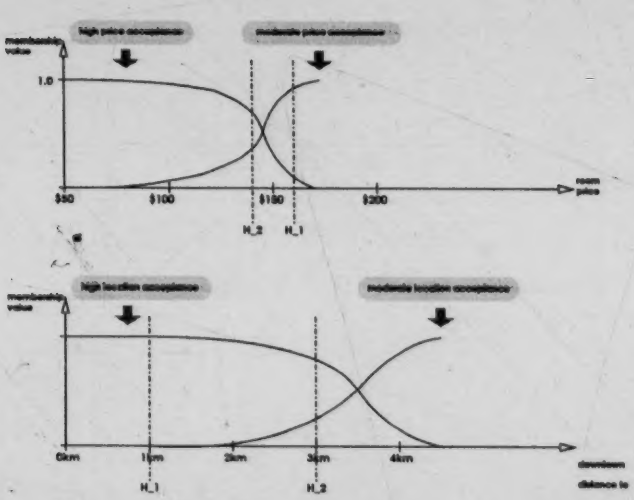


Figure 3.14: User's acceptance membership functions combined with hotel data

room price is higher than \$140. However, usage of fuzzy membership functions and approximate reasoning brings different outcome. The rules that govern the inference process are defined by the user, as in Table 3.4. The results of the approximate reasoning for such set of data are presented in Figure 3.15. It can be seen that the hotel #2 has obtained a slightly higher value than for the hotel #1 – the fact of being very close to the city centre has overcome the high price for a single room.

3.5.5 Service Example #2

This example is an extension of the previous one. This time service and facility preferences are taken into account. A set of experiments was performed in this case. In order to illustrate steps that are preformed during the process, a simple description of the most important aspects is shown below.

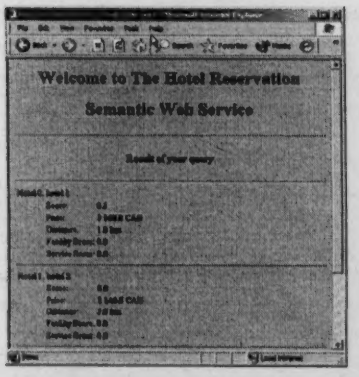


Figure 3.15: Result of service quarry for service example #1

A user is presented with a special menu to select his/her preferences regarding specific service and facility items, Figure 3.16. A user has three choices, and selections done by him/her are used to calculate acceptance membership functions shown in Section 3.4.2.2, as well as "scores" representing all items that a given hotel provides.

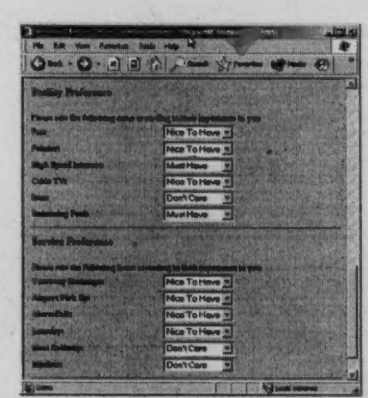


Figure 3.16: Dialog box for User's Preferences

These "scores" represent a matching between user's needs/preferences and hotel amenities. The calculations of "scores" is done according to the following approach. The selection done by the user, Figure 3.16, provides the system with information about times that are important for the user. The system "know" which item the user has to have, which ones he/she would like to have, and which ones are not even considered by the user. Each of three possibilities is associated with a weight, Section 3.4.2.2. The "score" is calculated based on the formula:

$$facility_score = \sum_{i}^{m} weigth_{nice} * facility_item_{nice,i} + \sum_{j}^{n} weigth_{must} * facility_item_{must,j}$$

where

$$facility_item_{nice,i} = \begin{cases} 0, \textit{ item is not provided by hotel} \\ 1, \textit{ item is provided by hotel} \end{cases}$$

$$facility_item_{must,j} = \begin{cases} -0.5, \textit{ item is not provided by hotel} \\ 1, \textit{ item is provided by hotel} \end{cases}$$

and m is a number of *nice to have* items and n is a number of *must have* items identified by the user. The same set of equations is used for calculation of the service score.

In our example, for one of the individuals, m and n are equal to 3 and 2 respectively for the facility selection, and 4 and 0 for the service selection. The analysis of preferences for such a scenario leads to the $facility_score$ of 6.0 and $service_score$ of 0.0 for the hotel #1, and 12.0 and 9.0 for the hotel #2 respectively. These scores together with room prices and localization information are subject of acceptance analysis. Degrees of "satisfaction" with all inputs are then passed to the inference engine. The engine identifies the overall acceptance of each service. In the case of this example, the acceptance of the hotel #1 is 2.5, and of the hotel #2 is 7.0, Figure 3.17.

Figure 3.17: Result of service query for service example #2

A set of experiments has been conducted with a number of individuals. It has to be stated that in the overwhelmed majority of cases the selection performed by the systems perfectly matched the selection done by a human. There were few cases of mismatch. We did discuss these mismatches with individuals for whom this happened.

An interesting conclusion is that they were not consistent during the selection process. Once they identified their acceptance levels, the selection process was performed with a slightly different acceptances. It seems that it is a feature of human nature to quickly change preferences and levels of acceptance regarding these preferences when confronted with a set of alternatives in real scenario.

Table 3.1: Information about hotels used in examples #1 and #2

	hotel_1	*hotel_2*
room price	$160.00	$140.00
distance to the city centre	1.0km	3.0km
fax	y	y
printer	–	y
high speed internet	y	y
cable TV	–	y
iron	y	y
swimming pool	–	–
currency exchange	–	y
airport pick up	–	–
alarm clock	–	y
laundry	–	y
meal delivery	–	y
minibar	–	y

Table 3.2: User's choices regarding facility items

Fax	*nice to have*
Printer	*nice to have*
High Speed Internet	*must have*
Cable TV	*nice to have*
Iron	*don't care*
Swimming Pool	*must have*

Table 3.3: User's choices regarding services

Currency Exchange	*nice to have*
Airport Pick Up	*nice to have*
Alarm Clock	*nice to have*
Laundry	*nice to have*
Meal Delivery	*don't care*
Minibar	*don't care*

Table 3.4: Matrix of fuzzy rules representing service acceptance (with two inputs: price and location acceptance)

	location acceptance		
price acceptance			
	low	*moderate*	*high*
low	low	low	moderate
moderate	moderate	moderate	high
high	moderate	high	high

Chapter 4

Computing with Words based Systems with Fuzzy Ontology

4.1 Motivation

A generalized constraint, as defined in Computing with Words (CW), is represented as

$$X \; isr \; R. \tag{4.1}$$

A simple instance of this generalized constraint is a proposition *John is about 16 years old.* For any human, this simple sentence brings information that: John is a male, and John is young, and goes to a high school. For people who knows John better, this sentence "invokes" other information, for example, John's father, a city where John lives, details of a high school he goes to. This means that the semantic

of the sentence is quite rich and the sentence alone brings a lot of information with it. If a query is made about John - an answer can be accurate and meaningful.

The richness and flexibility of definitions and instances of ontologies have led to the idea of their utilization for representing propositions for CW purposes. In the proposed approach, a set of definitions of relations X *isr* R constitutes an ontology. This ontology contains definitions of variables X, as well as definitions of constraining relations R. Due to the fact that this ontology resembles an *Explanatory Database* [81], it is named an *explanatory ontology*.

An important advantage of utilization of ontology is that definitions of variables and constraining relations that are parts of Explanatory Ontology can be easily personalized. In this case, instances of definitions of concepts represent perceptions of a single person, or a group of people who share the same perceptions.

4.2 Background: Computational with Word by Zadeh

CW is associated with processing of natural language-based information, as well as knowledge acquisition, representation, and processing. CW is based on the application of "fuzzy sets and fuzzy logic to address issues of ambiguity and imprecision in everyday human activities and possibly in information processing of constructed intelligent systems" [68]. The fact that CW is able to deal with words and propositions

that do not represent crisp measurements but individual's perceptions of the real world is very unique and important. It brings many challenges and research topics that are essential for successful applications of CW.

The fundamental concept of CW is related to application of propositions expressed in a natural language. In this case, the knowledge is expressed in the form of a constraint on one or more of the implicit variables. The first phase of the CW methodology focuses on a translation of these propositions into a computer manipulable language. The second phase in the process is a goal directed manipulation of these propositions. This phase can be seen as a kind of inference process. This inference process is based on a constraint propagation mechanism. The result of this second phase is a proposition providing a constraint on a variable of interest. The final phase is a process of retranslation; here, a statement in a computer manipulable language is converted into an appropriate statement in natural language.

As it has been stated above, a constraint plays a pivotal role in CW. Zadeh introduced a concept of generalized constraint in the form:

$$X \; isr \; R. \tag{4.2}$$

The constraint has two components – R is a constraining relation, and X is the constrained variable. The symbol *isr* represents a variable that defines the way in

which R constrains X. Depending on the value of this variable, the role of R is determined. The values of r with their meanings are as follow:

e:	equal (abbreviated to =);
d:	disjunctive (possibilistic)
v:	veristic
p:	probabilistic
γ:	probability value
u:	usuality
rs:	random set
rfs:	random fuzzy set
fg:	fuzzy graph
ps:	rough set (Pawlak set)

For example, when $r = d$ then the constraint is disjunctive (possibilistic) and *isr* is abbreviated to *is* resulting in the expression X *is* R, where R is a fuzzy relation which constrains X by playing the role of the possibility distribution.

A collection of relations X *isr* R is called *explanatory database* (ED). The relations of ED are very generic - they define relations without specifying any details regarding their concrete utilization. When the specifics are given, then ED is said to be *instantiated* and is denoted EDI [79].

4.3 Related Work

In a short period of time the paradigm of CW has become a very important topic of ongoing research activities that touch many issues related to intelligent systems. Two edited volumes with papers describing the results of research in the area of CW and related issues have been already published [82] [68]. A substantial number of papers dedicated and related to CW appear in international journals and conferences every year.

CW-related research activities embrace a wide range of topics. One can find CW-related papers that look at the issues of natural language processing, approximate reasoning, and interfacing between user and a CW-based system. There are also papers that focus on linguistic aspects of CW, on numerical aspects of CW, as well as on issues of computational models. A lot of attention is also dedicated to the application of the CW paradigm in a number of different scenarios.

The issue of using a specific input formats for CW was tackled in the paper by Qiua [48]. The approach presented there looked at possibilities of inputting strings of words (probability distributions over input alphabet) to a number of computational models like probabilistic finite automata, probabilistic Turing machines, and probabilistic context-free grammars. A work related to similar issues was reported in [69] [74] where fuzzy finite automata, fuzzy Turing machines, fuzzy regular grammars,

and fuzzy contextfree grammars with input strings of words were investigated. In particular, the work was concerned with computation tractability theorems.

The computing techniques and a reasoning process are critical aspects of CW. These topics have been a main focus of a number of papers. Application of approximate reasoning techniques to CW have been raised in [14] [72]. An interesting discussion about qualitative reasoning was included in [15]. In [10] the authors looked at the application of fuzzy arithmetics instead of fuzzy logic as the main computational principle for CW. Another approach was related to an application of automata (see above) to perform some computations. A new kind of fuzzy automata whose inputs are strings instead of values has been introduced in [74]. There is also work dedicated to the issues of uncertainty [55], cognition [30], and computational semiotic [51].

An output generation process is also an important aspect of CW. The retranslation stage in the paradigm of CW can be formulated as a multicriteria decision problem [73]. This paper introduced a number of criteria that can be used in the process of selecting the retranslation.

Importance and suitability of the CW paradigm can be supported by its ability to solve some real-life problems. A number of applications has been already reported:

- for the automatic text documents categorization [83], and information retrieval [33] by offering better processing of subjective descriptors, and by providing the user with a tool of understandable language based on words [5];
- for representations of the popular group decision making rules by extending them to traditional fuzzy preference relations [31];
- for designing fuzzy controllers by applying fuzzy Lyapunov synthesis, which is a computing with words version of classical Lyapunov synthesis [37] [87];
- for classification purposes via translating the natural language descriptions for bone age assessment [2];
- for measuring the information quality of Web sites and generating linguistic recommendations [26].

There is also work dedicated to development of suitable knowledge representation models. A new (proportional) 2-tuple fuzzy linguistic representation model for CW, which is based on the concept of "symbolic proportion" was proposed in [70]. The linguistic information was represented by means of 2-tuples, which are composed by two proportional linguistic terms.

Fuzzy conceptual graphs as a knowledge representation language were proposed in [6]. Fuzzy conceptual graphs were formulated as a generalization of conceptual graphs where fuzzy types and fuzzy attribute-values are used in place of crisp types

and crisp attribute-values. Projection and join as basic operations for reasoning on fuzzy conceptual graphs were defined, taking into account the semantics of fuzzy set-based values.

The issues of rule extraction, knowledge representation, and approximate reasoning based on Type 2 formulas for CW were highlighted in [62]. In [63], the proposal for using Type 2 fuzziness for knowledge representation and approximate reasoning for CW was further described. The emphasis was put on the ability to capture varying degrees of meaning for words, and to generate Fuzzy Disjunctive and Conjunctive Canonical Forms.

There are also few papers dedicated to the topic of building CW-based systems and frameworks supporting their development. One of these paper is [67], where a linguistic dynamic systems for CW was built by fusing procedures and concepts from several different areas: Kosko's geometric interpretation of fuzzy sets, Hsu's cell-to-cell mappings in nonlinear analyzis, equi-distribution lattices in number theory, and dynamic programming in optimal control theory. Among other three papers related to this topic [24], [34], and [22], the last one is of a special interest. It proposed a framework that was built as a network of objects that contained generic and fuzzy objects with some interaction among them.

Different from all the work mentioned above, the work done in this thesis enhances capabilities of performing CW by merging it with an ontology. A very important and innovative aspect of this approach relies on the fact that an ontology allows for representing semantics of words via definitions of concepts and different types of relations among them.

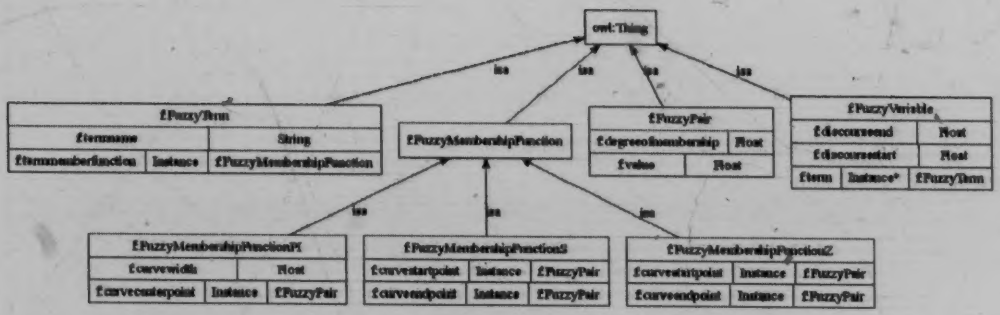
4.4 Computing with Words based System

4.4.1 Construction of Explanatory Ontology

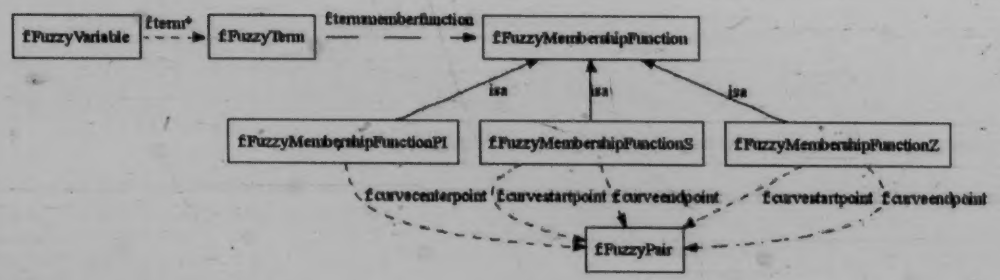
4.4.1.1 Relation Ontology

In the proposed approach, each type of R (eq. (4.2)) is described by a single ontology, called a constraining ontology. This ontology defines concepts existing in a given type of R, together with relationships existing among these concepts. This constraining ontology is built using a relation ontology. In the paper, a simple fuzzy relation ontology is used. Its core is shown in Fig. 4.1.

Fig. 4.1 a) illustrates a structure of the ontology. The concepts are structured according to their superclass-subclass relations. Fig. 4.1 b), on the other hand, focuses on links among the concepts. In overall, Fig. 4.1 shows a definition of the term *f:FuzzyVariable*. Its properties contain two datatypes that define a range of universe of discourse (*f:discoursestart* and *f:discourseend*), and one object prop-



(a)



(b)

Figure 4.1: Fuzzy Relation Ontology - Definition: (a) a view representing *isa* relationships among the concepts; (b) a view illustrating a single concept *FuzzyMembershipFunction* with some of its subclasses and relations with other concepts

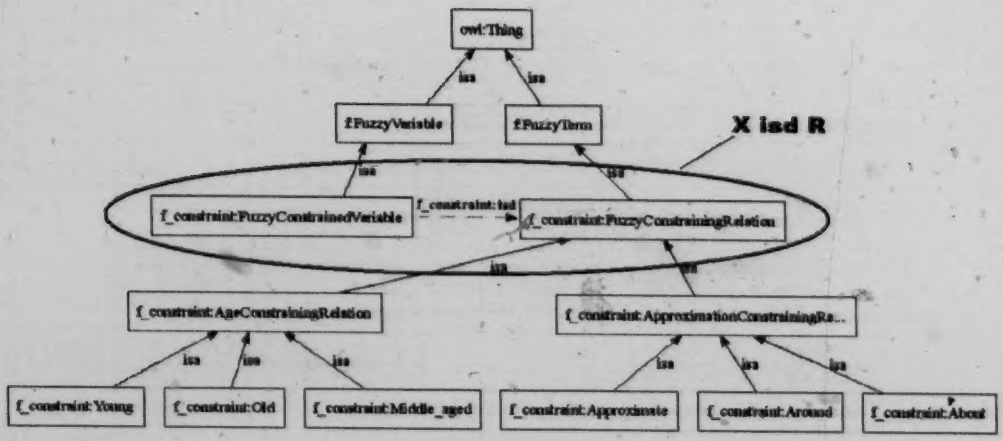


Figure 4.2: Fuzzy Constraining Ontology - Definition

erty *f:term* that points to the concept *f:FuzzyTerm*. The definition of *f:FuzzyTerm* points to the concept *f:FuzzyMemebershipFunction*. This class is a superclass for three other classes: *f:FuzzyMemebershipFunctionZ*, *f:FuzzyMemebershipFunctionPI*, and *f:FuzzyMemebershipFunctionS*. Each of these classes has properties that identify characteristic points of a given fuzzy membership function.

This simple example shows how a fuzzy variable can be created. An instance of it "needs" a number of fuzzy terms, and each fuzzy term "needs" a name (linguistic label) and a membership function associated with it.

4.4.1.2 Constraining Ontology

A relation ontology is a starting point for construction of a constraining ontology. The relation ontology is extended to become a constraining ontology. This process

means that all the terms and concepts defined by the relation ontology can be used and enhanced during development of constraining ontology. The type of relation ontology used for this process determines the type of constraining ontology.

Fig. 4.2 presents an example of a simple fuzzy constraining ontology that has been constructed based on the fuzzy relation ontology (Fig. 4.1). It can be seen that two fuzzy constraining relations has been defined there (*f_constraint: AgeConstrainingRelation* and *f_constraint:Approximation ConstrainingRelation*). Each of them has three subclasses, for example, *f_constraint: AgeConstrainingRelation* has the concepts *f_constraint: Old*, *f_constraint: Mid_aged*, and *f_constraint: Young*. It has to be said that all the concepts of the last row (Fig. 4.2) are subclasses of the concept *f:FuzzyTerm*. It means that each of them is linked to a fuzzy membership function.

Fig. 4.2 shows the way generalized constraints are defined. There is also the concept *f_constraint:FuzzyConstrainedVariable* in the Figure. This concept represents the component X of a constraint (eq. (4.2)). It has a object property *f_constrained: isd* that links it with *f_constraint:FuzzyConstrainingRelation* (and any of its subclasses). Such arrangement is shown in Fig. 4.2 in the oval.

4.4.1.3 Explanatory Ontology

A constraining ontology can be used to extend any domain specific ontology in order to build an explanatory ontology. Multiple ontologies containing terms and concepts related to any area of human's life and activity can be used here. In this way, many relations and context dependencies defined by these ontologies become a part of explanatory ontology.

The explanatory ontology presented in the paper is built based on the person ontology (Fig. 2.2). This ontology is extended by combining it with the fuzzy constraining ontology (Fig. 4.2). The resulted explanatory ontology is presented in Fig. 4.3. It can be seen that the property *ED:age* is defined differently. In the original person ontology, the property *ED:age* was of type float, now *ED:age* is represented as an object property.

It is important to understand the diagram presented in Fig. 4.3. The concept *ED:Age* is a subclass of the concept *f_constraint:FuzzyConstrainedVariable* that is a subclass of *f:FuzzyVariable* (see Fig. 4.2). This means that it has a range - *f:discoursestart* and *f:discourseend*, and a link to the concept *f_constraint: FuzzyConstrainingRelation*. The concept *f_constraint: FuzzyConstrainingRelation* (a subclass of *f:FuzzyTerm*) is a superclass of a number of possible constraining relations, see Fig. 4.2. This means that the concept *ED:Age* has an object property *f_constraint:isd* rep-

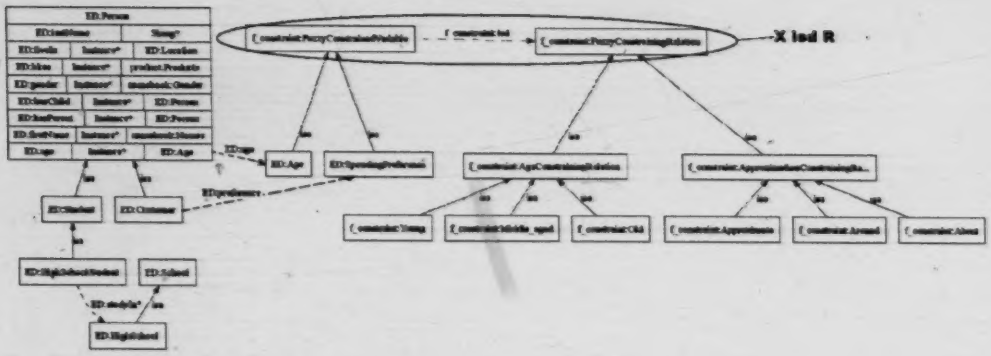


Figure 4.3: Explanatory Ontology - Definition

resenting a link to any subclass of *f_constraint: FuzzyConstrainingRelation* (the last row of nodes in Fig. 4.2).

The instance of the Explanatory Ontology is presented in Fig. 4.4. This ontology instance represents a similar information as the instance presented in Fig. 2.3. However, the age of John is presented as a fuzzy constraint relation. The object property *ED: age* of the instance *John* points to *JohnAge* which is the instance of the concept *ED: Age*. Further, the concept *JohnAge* has a link, *f_constraint: isd* to the instance of the node *ED: about_16* of the concept *f_constraint:About* that is a subclass of the concept *f_constraint: ApproximationConstrainingRelation* (Fig. 4.2).

This relationships define the constraint "*JohnAge isd about_16*". Fig. 4.4 contains one more constraint of the type *X isd R*. It is a constraint "*JohnAge isd young*". In

this case, the instance *JohnAge* is linked via object property *f_constraint: isd* with the instance *ED:young*. The *ED:young* is the instance of the concept *f_constraint:Young*, and is described by a membership function *ED:young_mf* of a type Z. See Fig. 4.4 for more details about the constrains *X isd R*.

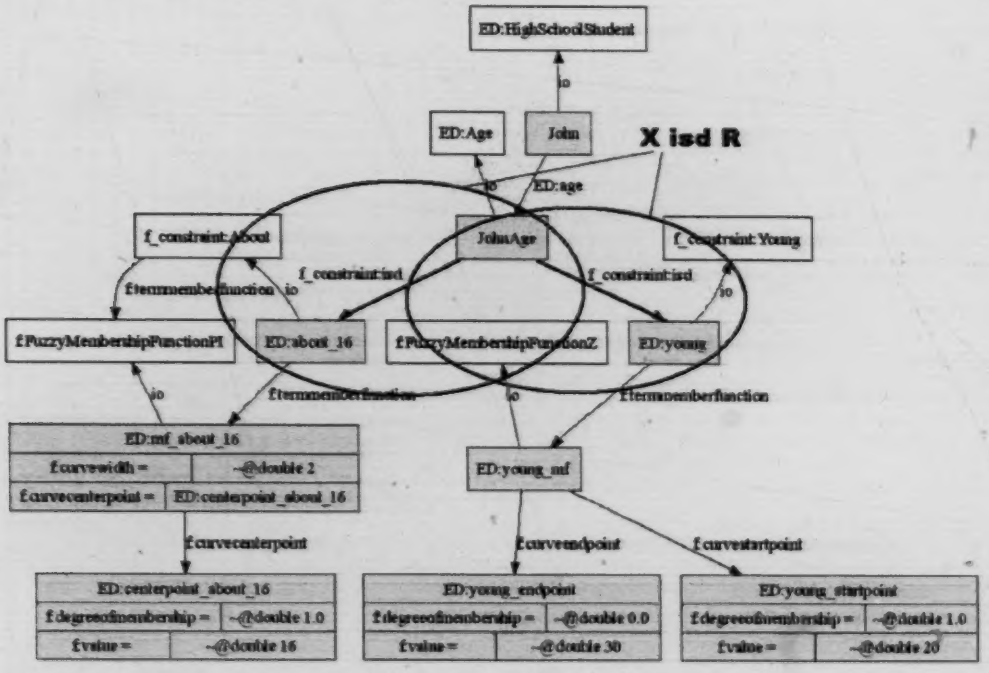


Figure 4.4: Explanatory Ontology - Instance (white boxes represent definitions of concepts, gray boxes represent instances)

4.4.2 Population of the Explanatory Database Ontology Instance

There are two different ways to populate instance of the ED. The first and simple way is to use web forms to ask for user's information, then directly fill the information into the EDOI. The disadvantage of this approach is there could be many forms the user would have to fill, and some of them might not even relevant to a specific user. However this approach is good for acquiring complicate information which the second approach can not handle; for example, the user's spending preference in fuzzy terms: "I would like to spend about $50"; a web form with graphic would help the user describe what he means by "about $50" much easier.

The second way to populate the instance of the EDO is using natural language processing. The users provide their information in form of natural language, English sentences in particular, the system will then translate those sentences into Ontology knowledge and insert the knowledge into the EDOI. The translation process could be done by integrating ontology with a NLP engine such as Lexical Knowledge Builder (LKB).

4.4.3 Implementation of the Computing with Words based System

4.4.3.1 Architecture and Behavior

An architecture of the CW-based system is presented in Fig. 4.5. The main components of the system are: Explanatory Ontology, Inference Engine, Personalization Unit, Input and Output Interfaces.

The Explanatory Ontology contains information and knowledge known to the system. The ontology, as presented in Fig. 4.5, is able to represent different types of general constraints. The type of general constraints depends on the type of used relation ontology and constraining ontology (Section 4.4.1). It is possible to build ontology that has multiple types of constraints. Besides definitions of concepts, their instances, the Explanatory Ontology also contains rules that are built based on these definitions and instances. These rules can be used to infer, based on information and knowledge already stored in the ontology, about new pieces of knowledge. Additionally, the system can have rules that are specific to the selected relation ontology. The Explanatory Ontology is populated by facts entered to the system, as well as by facts that have been deduced based on existing facts and rules.

The next important component of the system is the Inference Engine. As it can

be seen in Fig. 4.5, it can be built using a number of different reasoners. The generic reasoner is used to infer new facts based on precise information and generic rules. Information and knowledge expressed in approximate form, as well as perceptions, are used by specific reasoners that perform their tasks using relation specific rules.

The Inference Engine should constantly "monitor" information and knowledge stored in the Explanatory Ontology, and be able to deduce new facts. Both components of CW-based system - the Explanatory Ontology and the Inference Engine - are "interacting" with each other.

The core of the Inference Engine is the Fuzzy Reasoner introduced in Section 2.7

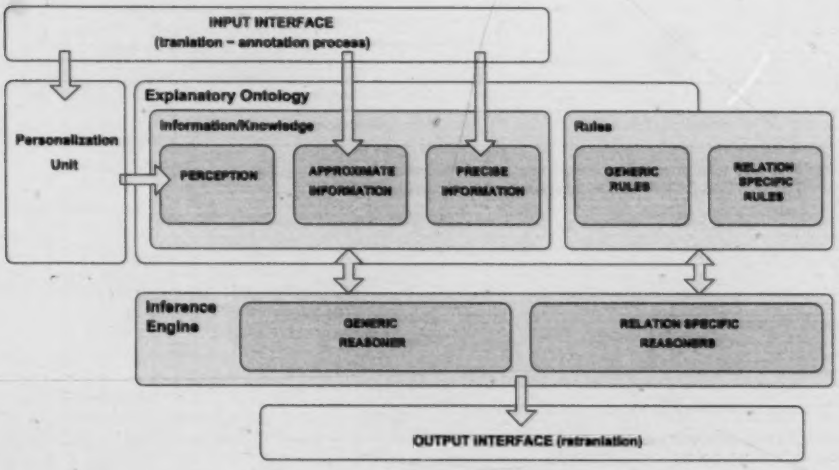


Figure 4.5: Explanatory Ontology based system for CW

The other three components: the Personalization Unit, Input and Output Interfaces are very important from the point of view of a user. However, the work associated with development of these components is related to separate research activities. The scope of the work in this thesis does not include them. The only note we would like to make about Input Interface is that in order for a ontology-based system to work, the Input Interface has to perform annotation of the input with ids of concept definitions and instances that already exits in the Explanatory Ontology.

4.4.4 Experiments and Results

In order to illustrate a suitability of application of the proposed Explanatory Ontology to build a CW-based system, a simple prototype of such a system has been constructed. The example included here illustrates a scenario when a simple new fact about a boy named John: *John studies in Queen Elizabeth* is entered into the system. This information alone leads to generation of a number of new facts about him. Additionally, this fact allows a system to answer a non-trivial question: *What to buy for John?* that is asked by two people that can spend different amounts of money on gifts.

4.4.4.1 Explanatory Ontology

The Explanatory Ontology built for the purpose of the prototype contains a number of ontologies:

- a fuzzy relation constrain ontology:

  the core concepts of fuzzy relation ontology are presented in Fig. 4.1, the ontology contains definitions of such concepts are *f:FuzzyTerm*, *f:FuzzyVariable*, *f:FuzzyMembershipFunction*, and *f:FuzzyPair*; these concepts are used in a fuzzy constraining ontology to built other concepts; the fuzzy relation ontology is used to define constrained variables - elements X (eq. (4.2)), as subclasses of the concept *f:FuzzyVariable*, for example the concept *ED:Age* is defined as the subclass of the *f_constraint:FuzzyConstrainedVariable*, the concept *ED:Age* is linked via the object property *f_constraint:isd* to the *f_constraint: AgeConstrainingRelation* (see Fig. 4.3);

- a fuzzy constraining ontology:

  a part of this ontology is presented in Fig. 4.2, it provides means to address two challenges: approximation and personalization; this ontology is used to define constraining relations - element R - in the equation X *isd* R, the concepts *f_contraint: AgeConstrainingRelation* and *f_contraint: Approximation-ConstrainingRelation* are defined here as subclasses of the concept *f:FuzzyTerm*, additionally the concepts *f_contraint: Young*, *f_contraint: Middle_aged*, *f_contraint: Old*, and *f_contraint: About*, *f_contraint: Around*, and *f_contraint: Approximate* are defined there too; the fact that these concepts are subclasses of the concept *f:FuzzyTerm* means that each of them is linked with its own membership function, parameters of these functions reflect user's unique perception of these

concepts;

- specific ontologies

  these ontologies represent information/knowledge about specific topics related to different aspects of environment; in the case of this example we use the following ontologies:

  - a location ontology - contains definitions of concepts *location:Country*, *location:Province_State*, and *location:City*;
  - a school ontology - contains concepts describing different types of schools, concepts *school:ElementarySchool*, *school:JuniorHighSchool*, *school:High School*, *school:University*;
  - a game ontology - it is an ontology that defines the following concepts: *product:Game*, and two subclasses *product:GameConsole*, and *product: VideoGame*.
  - a namebook ontology - it contains information about names that can be given to males and females - the concept *namebook:Gender*, and the concept *namebook:Name* with two subclasses *namebook:FemaleName* and *namebook:MaleName*.

- ontology-based rules;

  there are a number of rules that are built based on concepts and relations defined in the ontology, these rules provide a means to represent more complex relationship between concept properties;

- relation specific rules;

  there are a few rules that are created to deal with issues that are related to fuzzy reasoning.

4.4.4.2 Known Knowledge

Before the details of interaction with the sample CW-based system are shown, there is a need to present the information that the Explanatory Ontology already contains. The following pieces of information are expressed in the ontology:

- "simple" facts (instances of ontology concepts), examples of these facts are:
  - Edmonton, London, San Francisco - instances of *location:City*;
  - Canada, United Kingdom, United States - instances of *location:Country*;
  - John, Javis, Joelle - instances of *ED:Person*;
  - Queen Elizabeth - an instance of *school:HighSchool*;
  - Xbox360, PlayStation2, GameBoyAdvance - instances of *product:Game Console*;
  - Iron Phoenix, NHL 2006, BeatDown_FistofVengeance - instances of *product:VideoGames*;
- "complex" facts (represent relations between ontology instances), an examples is:
  - Edmonton is in Canada - a relation of *In(City, Country)*;

- approximation-related terms/concepts that are used to build X *isr* R constraints:
  - fuzzy constrained variables - *JohnAge*, *Joelle_spendingPreference*, *Javis_spendingPreference* (instances of concepts *ED:Age*, *ED:SpendingPreference*);
  - fuzzy constraining relations - *ED:young*, *ED:about_16*, *about_$300*, *about_$5* (instance of concepts *f_constraint:Young*, *f_constraint:About*);
- implicit and deduced facts (automatic extraction of information):
  - Queen Elizabeth is a High School - due to the fact that Queen Elizabeth is an instance of High School;
  - John is male - the result of deduction (RULE_A below);

4.4.4.3 Rules

The rules that are used in the example are of two types: the rules that are built based on the concepts defined in the Explanatory Ontology, and the rules that are related to the selected relation. The following rules are part of the ontology:

- RULE_A:

  ED:Person(?x) $\wedge$ ED:firstName(?x, ?n) $\wedge$ namebook:forGender(?n, ?g)

  $\rightarrow$ namebook: gender(?x, ?g)

- RULE_B:

  ED:Person(?x) $\wedge$ namebook:forGender(?x, Male) $\wedge$ ED:Age(?a1)

∧ ED:age(?x, ?a1) ∧ f_constraint:isd(?a1, young) ∧ product: Games(?g)

→ ED:likes(?x, ?g)

- RULE_C:

  ED:Person(?x) ∧ school:HighSchool(?h) ∧ location:City(?c)

  ∧ ED:studyIn(?x, ?h) ∧ ED:Location(?h, ?c)

  → ED:liveIn(?x, ?c)

- RULE_D:

  ED:Person(?x) ∧ ED:HighSchool(?h) ∧ ED:studyIn(?x, ?h)

  ∧ ED:Age(?a1) ∧ ED:age(?x, ?a1)

  → f_constraint:isd(?a1, ED:about_16)

The fuzzy related rules are:

- RULE_E:

  A isd B ∧ B is subset of C

  → A isd C

  (if A is represented by set B and set C contains set B then A is also represented

  by set C)

- RULE_F:

  A buys gift for B ∧ A has spending preference X

  ∧ B likes C ∧ C meets spending preference X

  → A is recommended to buy C

- RULE_G:

  price of C has membership value >= 0.5 of the fuzzy set representing the spending preference X

  → C meets spending preference X

4.4.4.4 Part I - New Information

The input information is the sentence *John studies in Queen Elizabeth*. This single statement triggers "firing" of some rules and leads to a few new facts:

- John lives in Edmonton - the result of deduction (RULE_C);
- John is about_16 - the result of deduction (RULE_D);
- John is young - the result of deduction (RULE_E);
- John likes games - the result of deduction (RULE_B), this also leads to the fact that John likes all the instances of the the concept *Games*.

4.4.4.5 Part II - Query

The next part of the example is about a question *What to buy for John?* that is asked by two relatives of John: Javis who lives in San Francisco, and Joelle who lives in London. Both of them are defined via the Explanatory Ontology. An example of the instance that defines Joelle is presented in Fig. 4.6. The most important object property of the instance *Joelle* of the concept *ED:customer* (relevant to the example) is a

link *ED:preference* that represents a triple $< ED : Person, ED : preference, ED : SpendingPreference >$. The *ED:Person* is *Joelle*, and *ED:SpendingPreference* is *Joelle_spendingPreference*. As we can see in Fig. 4.6, the instance *Joelle_spending-Preference* is the component X of the generalized constraint X *isd* R with the instance *about_300_dollars* of the constraining relation *f_constraint:About*. So, it represents a constraint "*Joelle_spendingPreference isd about_300_dollars*".

Once the inference processed occurred, new instances and relations were created. The state of the ontology after that is presented in Fig. 4.7. Comparing it with Fig. 4.6, we can say that a new link has been created. The link *ED:isRecommendedToBuy* that links *Joelle* with a suggested product to buy - the instance *product: xbox360*.

The instance that "defines" Javis is very much the same except the difference that the generalized constraint is the instance *about_50_dollars*.

In overall, when both Joelle and Javis asked the same question *What to buy for John?* the system responded differently. In the case of Joelle, the system pointed to the game component with the price of $299.99 - Xbox360. For Javis, the response was the game *BeatDown_FistofVengeance* for $29.99. The fragment of the ontology representing *John* after all information was entered, and a question was asked is presented in Fig. 4.8. The most interesting fact is the expansion of the instance *John*

that has a number of object links connecting *John* to instances of the ontology *product*: *product:xbox360*, *product:NHL_2006*, *product:playstation2*, and so on. Fig. 4.8 contains also indications about constrains X *isd* R, and induced links (dashed lines).

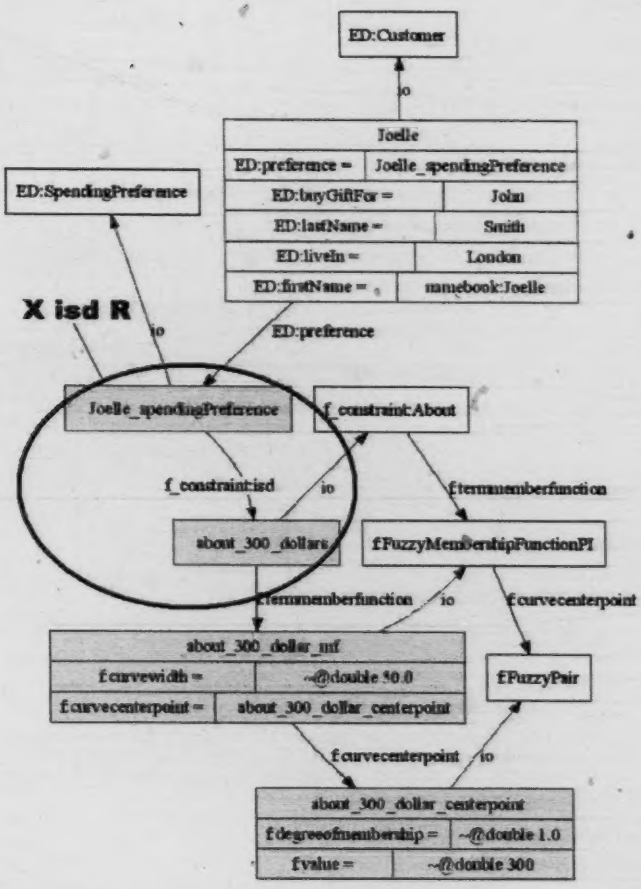


Figure 4.6: The instance Joelle - a snapshot of the ontology BEFORE the inference process occurred (white boxes represent definitions of concepts, gray boxes represent instances)

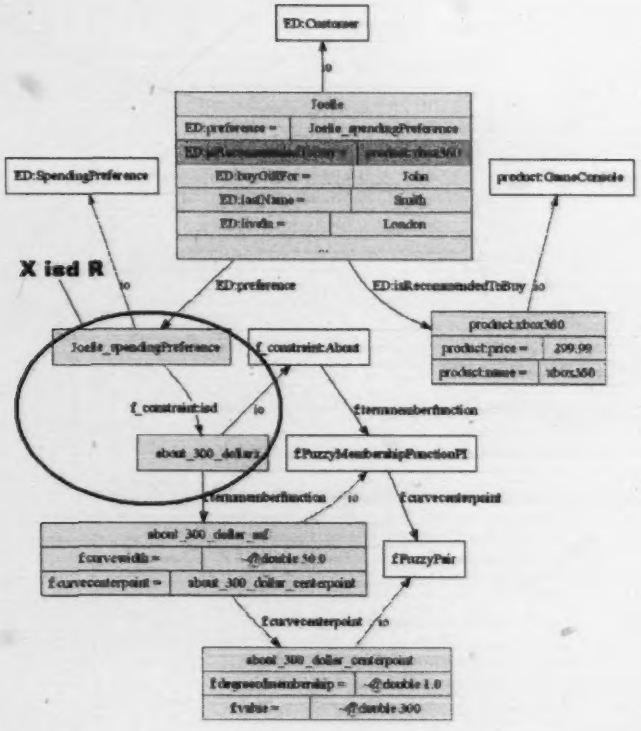


Figure 4.7: The instance Joelle - a snapshot of the ontology AFTER the inference process occurred (white boxes represent definitions of concepts, gray boxes represent instances)

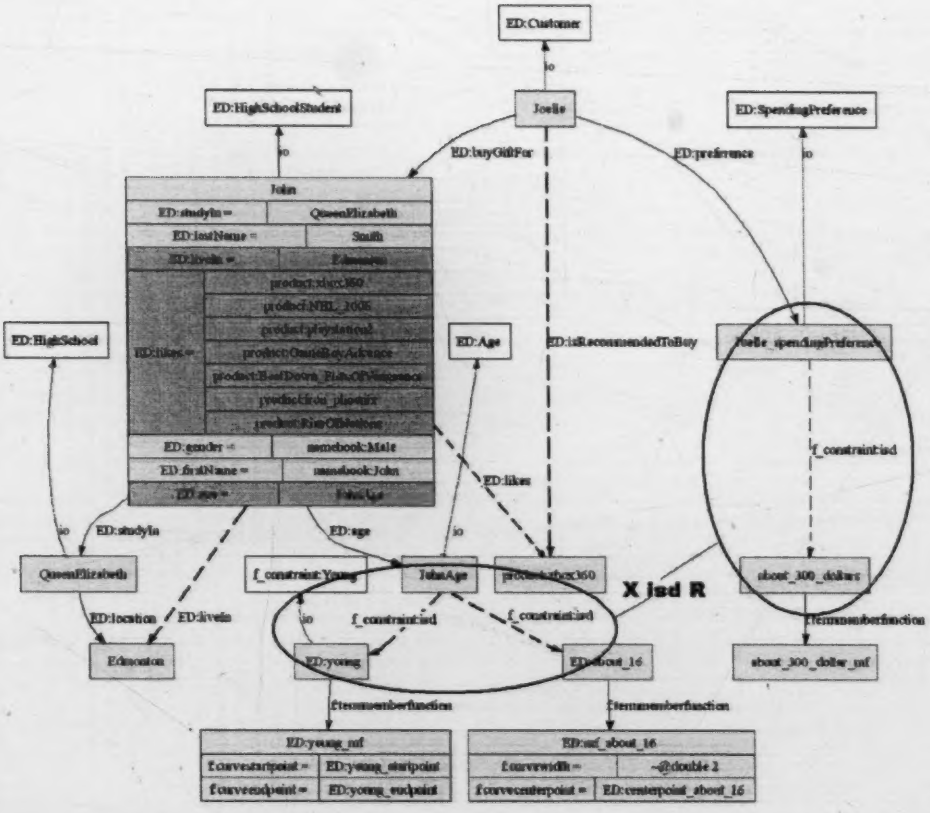


Figure 4.8: Explanatory Ontology after the inference process (white boxes represent definitions of concepts, gray boxes represent instances)

Chapter 5

Conclusions

5.1 Contributions

Research activities related to Human-Centered Computing draw a lot of attention among researchers in the fields of computer science and engineering. The results of these activities are very quickly implemented. Nevertheless, the computers still lack the capabilities to comprehensively communicate with and understand human beings.

The work presented in this thesis proposes a fuzzy ontological approach that allows development of computer systems with ability to imitate human behavior. Such systems are capable of learning more about the user by implying new facts about him/her based on information representing human behavior and already known facts. This concept is used as the basis for building fuzzy ontology, and the frameworks for

development of systems for Human-Centered Services and Computing with Word.

The creation of fuzzy ontology allows vague information to be expressed in a structured and meaningful way that is understood by both human and machine. This simplifies the communication process between human and machine. Using this ontology, user can describe his/her needs in natural linguistic terms. The fuzzy ontology introduced in this work is one of a few existing practical fuzzy ontologies in which it is not only capable of describing entities using fuzzy concepts, but also built to be easily processed by reasoning engines.

Using the fuzzy ontology, the Human-Centered Service system is able to gather and store even vague or incomplete information about the user. It then analyzes the information to learn about user needs, and builds a user acceptance profile that closely represents behavioral patterns of the user. Based on this profile, the service system performs different service-related tasks on behalf of its user. The user acceptance profile is an unique idea that has not seen anywhere else. It is created based on the perception that human user may prefer one thing but is willing to accept an alternative with a bit less satisfaction. For instance, a user specifies that he/she would like to stay in a 3-star hotel, but would accept a 2-star one.

As human-centered computer systems take human aspects as the core elements of

their activities they are required to "understand" their users and learn about them in the human way based on implicit knowledge. The Computing with Word based system is built to give the computer such capability. In this system, the concept of generalized constraints introduced by Zadeh is implemented. The explanatory ontology and its instance represent the explanatory database (ED) and the explanatory database instance (EDI). The reasoning with perceptions is performed by approximate reasoning engine using the fuzzy logic rules and the explanatory ontology with its instances.

These three elements described above constitute basic parts of our envisioned Human-Centered Computing system.

5.2 Future Work

The work done in this thesis focuses on presenting the idea of how fuzzy concepts and ontology can be combined together to create a human-centered system capable of learning, understanding user needs, and performing work on behalf of the user. Therefore, many details that would make these systems more complete and practical are left undone. For instance, the fuzzy ontology is created with only three types of membership functions: S, Z and Π. A more completed system should include other types of membership functions, as well as more mechanisms for fuzzy and approximate

reasoning. Similarly, the Human-Centered Service system only gathers information taken directly from the user. A more realistic system should be able to learn more about its user from a feedback provided by the user after a service is performed. Lastly, the Computing with Word system is built based on a limited explanatory ontology. A practical system should operate on a more carefully designed ontology which defines much more concepts.

Looking back at our vision of a Human-Centered Computing system, it can be said that a very important component dedicated to the speech recognition and Natural Language Processing has not been investigated. This could be the main topic of our future work. Our plan for this work would be to look for a speech recognition engine, such as Dragon NaturallySpeaking, capable of converting speeches into texts. Then, we would use a linguistic engine for deep language processing, such as Lexical Knowledge Base builder (LKB), to translate the converted texts into knowledge in the form of ontology. Having the speech recognition and natural language processing component in place our envisioned Human-Centered Computing system can be completed. The next step would be to apply this system to solve real life problem and improve it.

Bibliography

[1] "accept". Merriam-Webster Online Dictionary (2005). URL: http://www.merriam-webster.com

[2] S. Aja-Fernandez, R. De Luis-Garcia, M. A. Martin-Fernandez, and C. Alberola-Lopez, A computational TW3 classifier for skeletal maturity assessment. A Computing with Words approach, *Journal of Biomedical Informatics* **37** (2004) 99–107.

[3] T. Berners-Lee, D. Brickley, D. Connolly, M. Dean, S. Decker, P. Hayes, J. Heflin, J. Hendler, O.Lassila, D. McGuinness, and L.A. Stein, Reference description of the DAML+OIL (March 2001) ontology markup language, W3C World Wide Web Consortium, URL: http://www.w3.org/2001/09/06-ecdl/slide17-0.html

[4] T. Berners-Lee, J. Hendler, and O. Lassila, The Semantic Web, *Scientific American* **284** (2001) 34–43.

[5] F. Berzal, M. J. Martn-Bautista, M. A. Vila, and H. L. Larsen, Computing with words in information retrieval, *Annual Conference of the North American Fuzzy*

Information Processing Society - NAFIPS (2001) 3088–3092.

[6] T. H. Cao, A formalism for representing and reasoning with linguistic information,*International Journal of Uncertainty, Fuzziness and Knowlege-Based Systems* **10** (2002) 281–307.

[7] W.E. Combs, The Combs Method for Rapid Inference, in E. Cox (ed.) *The Fuzzy Systems Handbook* 2nd Edition. New York: AP Professional (1998) 659-680.

[8] The DARPA Agent Markup Language Homepage, URL: http://www.daml.org.

[9] S. Decker, F. van Harmelen, J. Broekstra, M. Erdmann, D. Fensel, I. Horrocks, M. Klein, and S. Melnik, The semantic web: The roles of XML and RDF, *IEEE Internet Computing* **4**(5) (2000) 63–74.

[10] M. Delgado, O. Duarte, and I. Requena, An Arithmetic Approach for the Computing with Words Paradigm, *International Journal of Intelligent Systems* **21** (2006) 121–142.

[11] M. L. Dertouzos. *The Unfinished Revolution: Human-Centered Computers and What They can do for us.* HarperCollins Publishers (2001).

[12] D. Dubois, and H. Prade. Fuzzy sets in approximate reasoning, Part 1: Inference with possibility distributions, *Fuzzy Sets and Systems* **40** (1991) 143–202.

[13] D. Dubois, and H. Prade. Fuzzy sets in approximate reasoning, Part 2: Logical approaches, *Fuzzy Sets and Systems* **40** (1991) 203–244.

[14] D. Dubois, L. Foulloy, S. Galichet, and H. Prade. Performing Approximate Reasoning with Words, in [82] 24–49.

[15] D. Dubois, A. Hadj-Ali, and H. Prade, Fuzzy Qualitative Reasoning with Words, in [68] 347–366.

[16] E. J. Friedman-Hill, *Jess in Action. Java Rule-based Systems* Manning Publications Co. (2003)

[17] D. Fensel, F. van Harmelen, I. Horrocks, D. L. McGuinness, and P. F. Patel-Schneider, OIL: An ontology infrastructure for the semantic web, *IEEE Intelligent Systems*, **16**(2) (2001) 38–45.

[18] M. Gao, and C. Liu, Extending OWL by Fuzzy Description Logic, *Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence* (2005) 562–567.

[19] S. Gasson, Human-Centered vs. User-Centered Approach to Information System Design, *Journal of Technology Theory and Application (JITTA)* **5**(2) (2003) 29–46.

[20] E. J. Glover, S. Lawrence, M. D. Gordon. W. P. Birmingham, and C. L. Giles. Recommending Web Documents Based on User Preferences, *Workshop on Recommender Systems: Algorithms and Evaluation* (1999).

[21] T. E. Gruber, A translation approach to portable ontology specifications, *Knowledge Acquisition* **5** (1993) 199–220.

[22] R. R. Gudwin, and F. A. Gomide, Object Networks: A Computational Framework to Compute with Words, in [82] 443–478.

[23] M. M. Gupta, Fuzzy-ism – The First Decade, in *Fuzzy Automata and Decision Processes* Elsevier. (1977)

[24] Y. Hattori, and T. Furuhashi, Study on a Framework for Solving Ill-defined Problems Using Patterns and Symbols, in [82] 479–494.

[25] P. Hayes, and B. McBride, RDF Semantics, W3C Recommendation 10 February 2004, URL: http://www.w3.org/TR/2004/REC-rdf-mt-20040210/.

[26] E. Herrera-Viedma, G. Pasi, A. G. Lopez-Herrera, and C. Porcel, Evaluating the information quality of Web sites: A methodology based on fuzzy computing with words, *Journal of the American Society for Information Science and Technology* **57** (2006) 538–549.

[27] I. Horrocks, DAML+OIL: a Description Logic for the Semantic Web, *IEEE Data Engineering Bulletin* **25**(1) (2002) 4–9.

[28] I. Horrocks, and P. F. Patel-Schneider, Proposal for OWL Rule Language, *13th International World Wide Web Conference* (2004) 723–731.

[29] Jena. URL: http://jena.sourceforge.net/

[30] B. A. Juliano, Cognitive Sciences and Computing with Words, in [68] 235–250.

[31] J. Kacprzyk, and S. Zadrozny, Computing with words in decision making through individual and collective linguistic choice rules, *International Journal of Uncertainty, Fuzziness and Knowlege-Based Systems* **9** (2001) 89–102.

[32] A. Kiss, and J. Quinqueton, Multiagent Cooperative Learning of User Preferences, *European Conference on Machine Learning/European Conference on Principles and Practice of Knowledge Discovery in Databases* (2001).

[33] B. Kostek, Computing with words concept applied to musical information retrieval, *Electronic Notes in Theoretical Computer Science* **82** (2003) 145–156.

[34] V. Kreinovich, B. Penn, and S. Starks, From Expert Words Directly to Numerical Simulations: Group-theoretic Approach to Computing with Words in Information/Intelligent Systems, in [82] 495–516.

[35] S. Agarwal, and S. Lamparter, User Preference Based Automated Selection of Web Service Compositions, *ICSOC Workshop on Dynamic Web Processes* (2005) 1–12.

[36] C. S. Lee, Z. W. Jian, and L.-K. Huang, A fuzzy ontology and its application to news summarization, *IEEE Transactions on Systems, Man and Cybernetics, Part B*, **35** (5) 859–880.

[37] M. Margaliot, and G. Langholz, Fuzzy control of a benchmark problem: A computing with words approach, *IEEE Transactions on Fuzzy Systems* **12** (2004) 230–235.

[38] D. L. McGuinness, and F. Van Harmelen, OWL Web Ontology Language Overview, W3C World Wide Web Consortium, URL: http://www.w3.org/TR/2003/PR-owl-features-20031215/

[39] E. Miller. Digital Libraries and the Semantic Web, W3C World Wide Web Consortium, URL: http://www.w3.org/2001/09/06-ecdl/slide17-0.html

[40] B. Ordhard, FuzzyJ, URL: ai.iit.nrc.ca/IR_public/fuzzy/fuzzyJToolkit.html (2002).

[41] J. Z. Pan, G. Stoilos, G. Stamou, V. Tzouvaras, and I. Horrocks, f-SWRL: A Fuzzy Extension of SWRL, *Journal of Data Semantics* **4** (2006) 27–45.

[42] M. Paolucci, K. Sycara, and T. Kawamura, Delivering Semantic Web Services, *The 12th International World Wide Web Conference* (2003)

[43] K. Sycara, D. Martin, D. L. McGuinness, S. McIlraith, and M. Paolucci, OWL-S Technology for Representing Constraints and Capabilities of Web Services, *W3C Workshop on Constraints and Capabilities for Web Services* (2004).

[44] M. Paolucci, A. Ankolekar, N. Srinivasan, and K. Sycara, The DAML-S Virtual Machine, *International Semantic Web Conference* (2003) 290–305

[45] J. Parsons, P. Ralph, and K. Gallagher, Using Viewing Time to Infer User Preference in Recommender Systems, *AAAI Workshop in Semantic Web Personalization* (2004).

[46] P. F. Patel-Schneider, P. Hayes, and I. Horrocks, OWL Web Ontology Language Semantics and Abstract Syntax, W3C Recommendation 10 February 2004, UML: http://www.w3.org/TR/2004/REC-owl-semantics-20040210/.

[47] W. Pedrycz, and F. Gomide, *An Introduction to Fuzzy Sets: Analysis and Design* MIT Press. (1998).

[48] D. Qiua, and H. Wang, A probabilistic model of computing with words, *Journal of Computer and System Sciences* **70** (2005) 176–200.

[49] L. Razmerita, A. A. Angehrn, and A. Maedche, Ontology-Based User Modeling for Knowledge Management Systems, *9th International Conference User Modeling 2003* Johnstown, PA, USA, (2003) 213–217.

[50] E. Reategui, J. A. Campbell, and R. Torres, Using Item Description in Recommender Systems, AAAI Workshop in Semantic Web Personalization, San Jose, California, (2004).

[51] B. B. Rieger, Computing Granular Word Meanings: A Fuzzy Linguistic Approach in Computational Semiotics, in [68] 147–208.

[52] F. Scott, W. D. Lewis, and D. T. Langendoen, An Ontology for Linguistic Annotation 1-2, *Fourteenth Innovative Applications of AI Conference* (2002) 11–19.

[53] B. Smith, Ontology. An Introduction, in L. Floridi (ed.), *Blackwell Guide to the Philosophy of Computing and Information*, Oxford: Blackwell, (2003) 155–166.

[54] M. K. Smith, C. Welty, and D. L. McGuinness. OWL web ontology language guide. W3C Recommendation, 10 February 2004, URL: http://www.w3.org/TR/owl-guide/.

[55] M. J. Smithson, Words about Uncertainty: Analogies and Contexts, in [82] 119–138.

[56] U. Straccia, Towards a Fuzzy Description Logic for the Semantic Web, *Second European Semantic Web Conference* (2005) 167–181.

[57] S. Staab, M. Erdmann, A. Maedche, and S. Decker, An Extensible Approach for Modeling Ontologies in RDF(S), *ECDL Workshop on the Semantic Web* **72** (2000) 234–253.

[58] G. Stoilos, G. Stamou, V. Tzouvaras, J. Z. Pan, and I. Horrocks, Fuzzy OWL: Uncertainty and the Semantic Web, *In Proc. of the International workshop on OWL: Experience and Directions* (2005).

[59] U. Straccia, *A Fuzzy Description Logic for the Semantic Web* ELSEVIER (2006).

[60] W3C World Wide Web Consortium, Semantic Web, URL: http://www.w3.org/2001/sw/

[61] Q. T. Tho, S. C. Hui, A. C. M. Fong, and T. H. Cao, Automatic fuzzy ontology generation for semantic Web, *IEEE Transactions on Knowledge and Data Engineering* **18** (6) (2006) 842 - 856

[62] I. B. Turksen, Computing with descriptive and veristic words: knowledge representation and approximate reasoning, in [68] 297–328.

[63] I. B. Turksen, Type 2 representation and reasoning for CWW, *Fuzzy Sets and Systems* **127** (2002) 17–36.

[64] Verberne A., van Harmelen F., and ten Teije A, Anytime Diagnostic Reasoning using Approximate Boolean Constraint Propagation, *Int. Conference on Principles of Knowledge Representation and Reasoning* (2000).

[65] W3C World Wide Web Consortium, Semantic Web News and Events Archive, URL: http://www.w3.org/TR/2003/PR-owl-features-20031215/

[66] W3C World Wide Web Consortium, OWL Web Ontology Language Overview, URL: http://www.w3.org/TR/owl-features/

[67] F.-Y. Wang, Y. Lin, and J. B. Pu, Linguistic dynamic systems and computing with words for complex systems, *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics* **4** (2000) 2399–2404.

[68] P. P. Wang (Ed.), *Computing With Words* Wiley-Interscience (2001).

[69] H. Wang, and D. W. Qiu, Computing with words via Turing machines: a formal approach, *IEEE Transactions on Fuzzy Systems* **11** (2003) 707–718.

[70] J.-H. Wang, and J. Hao, A new version of 2-tuple fuzzy linguistic representation model for computing with words, *IEEE Transactions on Fuzzy Systems* **14** (2006) 435–445.

[71] W. Wu, B. Parsia, E. Sirin, J. Hendler, and D. Nau, Automating DAML-S web services composition using SHOP2, in *Proceedings of 2nd International Semantic Web Conference* (2003).

[72] R. R. Yager, Approximate Reasoning as a Basis for Computing with Words, in [82] 50–77.

[73] R. R. Yager, On the Retranslation Process in Zadehs Paradigm of Computing With Words, *IEEE Transactions on Systems, Man, and Cybernetics, Part B* **34** (2004) 1184–1195.

[74] M. S.Ying, A formal model of computing with words, *IEEE Transactions on Fuzzy Systems* **10** (2002) 640–652.

[75] L. A. Zadeh, Fuzzy Sets, *Information and Control* **8** (1965) 338–353.

[76] L. A. Zadeh, A Fuzzy-Set Theoretic Interpretation of Linguistic Hedges, *Journal of Cybernetic* **2** (1972) 2–34

[77] L. A. Zadeh, The Concept of a Linguistic Variable and Its Application to Approximate Reasoning, *Information Science* (1975) **8** 199–249, 301–357 **9** 43-80

[78] L. A. Zadeh, A theory of approximate reasoning, *Machine Intelligence* **9** (1979) 149–194

[79] L. A. Zadeh, Test-score semantics of natural languages, *Proc. of 9th Int. Conf. on Computational Linguistics* Prague (1982) 425–430.

[80] L. A. Zadeh, Fuzzy logic = Computing with words, *IEEE Transactions on Fuzzy Systems* **4** (1996) 103–111.

[81] L. A. Zadeh, From computing with numbers to computing with words – from manipulation of measurements to manipulation of perceptions, *IEEE Transactions on Circuits and Systems – I: Fundamental theory and applications* **45** (1999) 105–119.

[82] L. A. Zadeh, and J. Kacprzyk (Eds.), *Computing with Words in Information/-Intelligent Systems 1.* Physica-Verlag Heidelberg (1998).

[83] S. Zadrozny, and J. Kacprzyk, Computing with words for text processing: An approach to the text categorization, *Information Sciences* **176** (2006) 415–427.

[84] L.A. Zadeh, The concept of linguistic variable and its application to approximate reasoning, Part 1, *Information Science* **8** (1975) 199-249.

[85] L.A. Zadeh, The concept of linguistic variable and its application to approximate reasoning, Part 2, *Information Science* **8** (1975) 301-357.

[86] L.A. Zadeh, The concept of linguistic variable and its application to approximate reasoning, Part 3, *Information Science* **9** (1976) 43-80.

[87] C. Zhou, D. Ruan, Fuzzy control rules extraction from perception-based information using computing with words, *Information Sciences* **142** (2002) 275–290.

[88] H. J. Zimmermann, Fuzzy Set Theory – and Inference Mechanism, in *Mathematical Models for Decision Support*. Springer-Verlag (1988).